# CH32X035 Reference Manual

V1.8

https://wch-ic.com

## Overview

CH32X035 is an industrial-grade microcontroller based on the QingKe RISC-V core. CH32X035 has built-in full-speed USB and USB PD PHY, supports USB Host and USB Device functions, USB PD and type C fast charging functions, built-in programmable protocol I/O controller (PIOC), provides 2 OPAs, 3 CMPs, 4 USARTs, I2C, SPI, multiple timers, 12-bit ADC, 14-channel touch-key, etc.

This manual provides detailed information on the use of the CH32X035 product for the development of user applications.

Please refer to the following datasheet for device characteristics of this family: *CH32X035DS0*.

For information about the RISC-V core, please refer to the following: *QingKeV4_Processor_Manual*.

**RISC-V core version comparison overview**

| Features / Core versions | Instruction set | Hardware stack levels | Interrupt nesting levels | Fast interrupt channels | Integer division cycles | Vector table model | Extended instruction | Memory protection |
|---|---|---|---|---|---|---|---|---|
| QingKeV4B | IMAC | 2 | 2 | 4 | 9 | Address or command | Supported | None |
| QingKeV4C | IMAC | 2 | 2 | 4 | 5 | Address or command | Supported | Standard |
| QingKeV4F | IMAFC | 3 | 8 | 4 | 5 | Address or command | Supported | Standard |

**CH32X035 product overview**

| CH32X035 | |
|---|---|
| QingKeV4C core | |
| 65K Flash | 20K SRAM |
| ADC(TKey)<br>2*ADTM<br>GPTM<br>4*USART<br>SPI<br>I2C<br>USBFS<br>USBPD<br>DMA<br>PIOC<br>2*WDG<br>2*OPA<br>3*CMP | |

**Abbreviated description of the bit attribute in the register:**

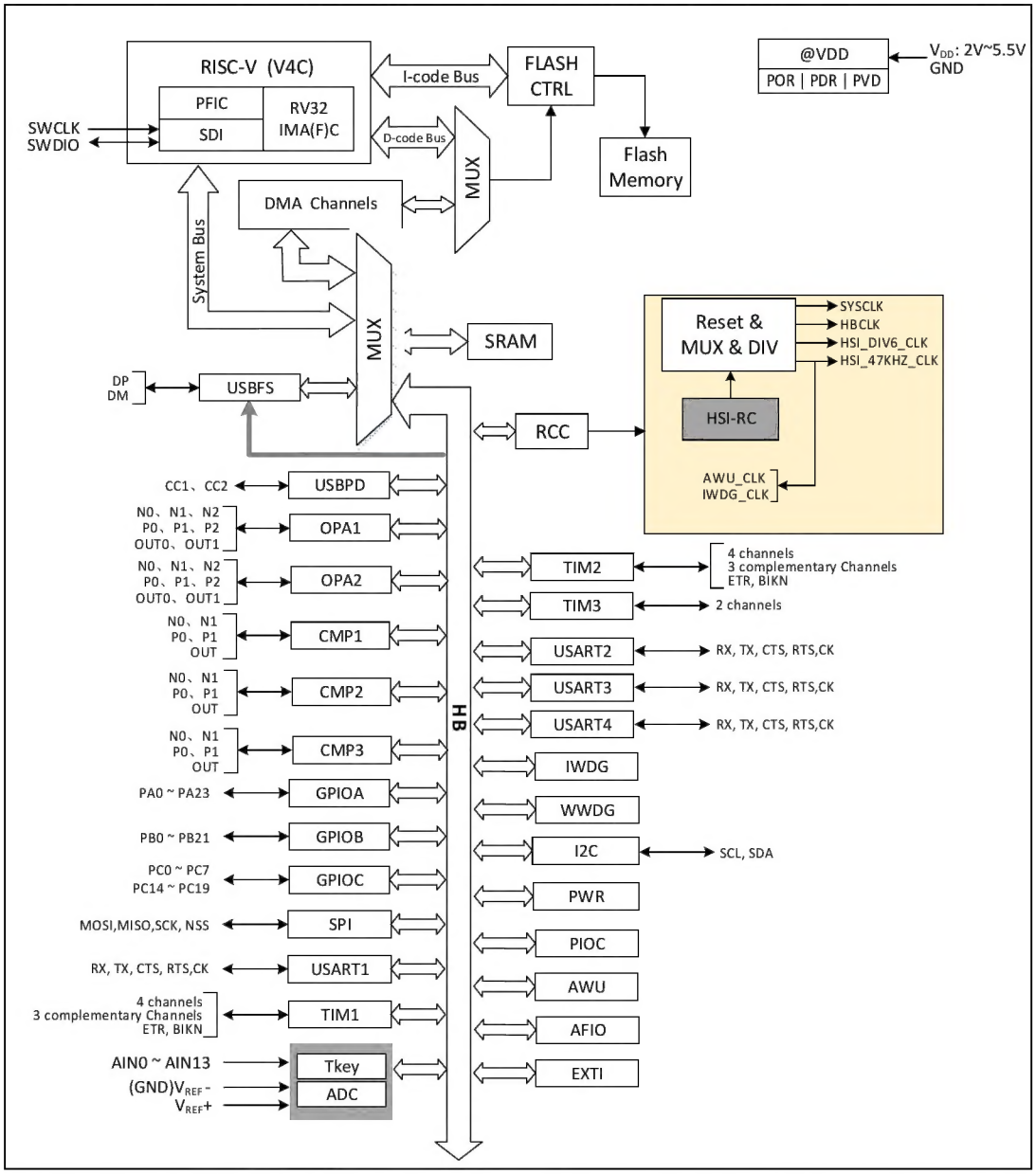
| Register bit attributes | Property description |
|---|---|
| RF | Read-only attribute, read a fixed value. |
| RO | Read-only attribute, changed by hardware. |
| RZ | Read-only attribute, auto bit clear 0 after read operation. |
| WO | Write only attribute (Not readable, read value uncertain) |
| WA | Write-only attribute, writable in Safe mode. |
| WZ | Write only attribute, auto bit clear 0 after write operation. |
| RW | Readable and writable. |
| RWA | Readable, writable in Safe mode. |
| RW1 | Readable, write 1 valid, write 0 invalid. |
| RW0 | Readable, write 0 valid, write 1 invalid. |
| RW1T | Readable, write 0 invalid, write 1 flipped. |
| SC | Auto cleared. |

# Chapter 1 Memory and Bus Architecture

## 1.1 Bus Architecture

The microcontroller is designed on the basis of the RISC-V instruction set and its architecture incorporates QingKe microprocessor core, arbitration unit, DMA module, SRAM storage and other components interacting via multiple sets of buses. Its system block diagram is shown in Figure 1-1.

Figure 1-1 CH32X035 system block diagram




The system is equipped with: General-purpose DMA controller to reduce the CPU burden and improve efficiency; clock tree hierarchy management to reduce the total power consumption of peripherals, as well as data protection mechanisms, clock security system protection mechanisms and other measures to increase system stability.

- The instruction bus (I-Code) connects the core to the FLASH instruction interface and prefetching is done on this bus.

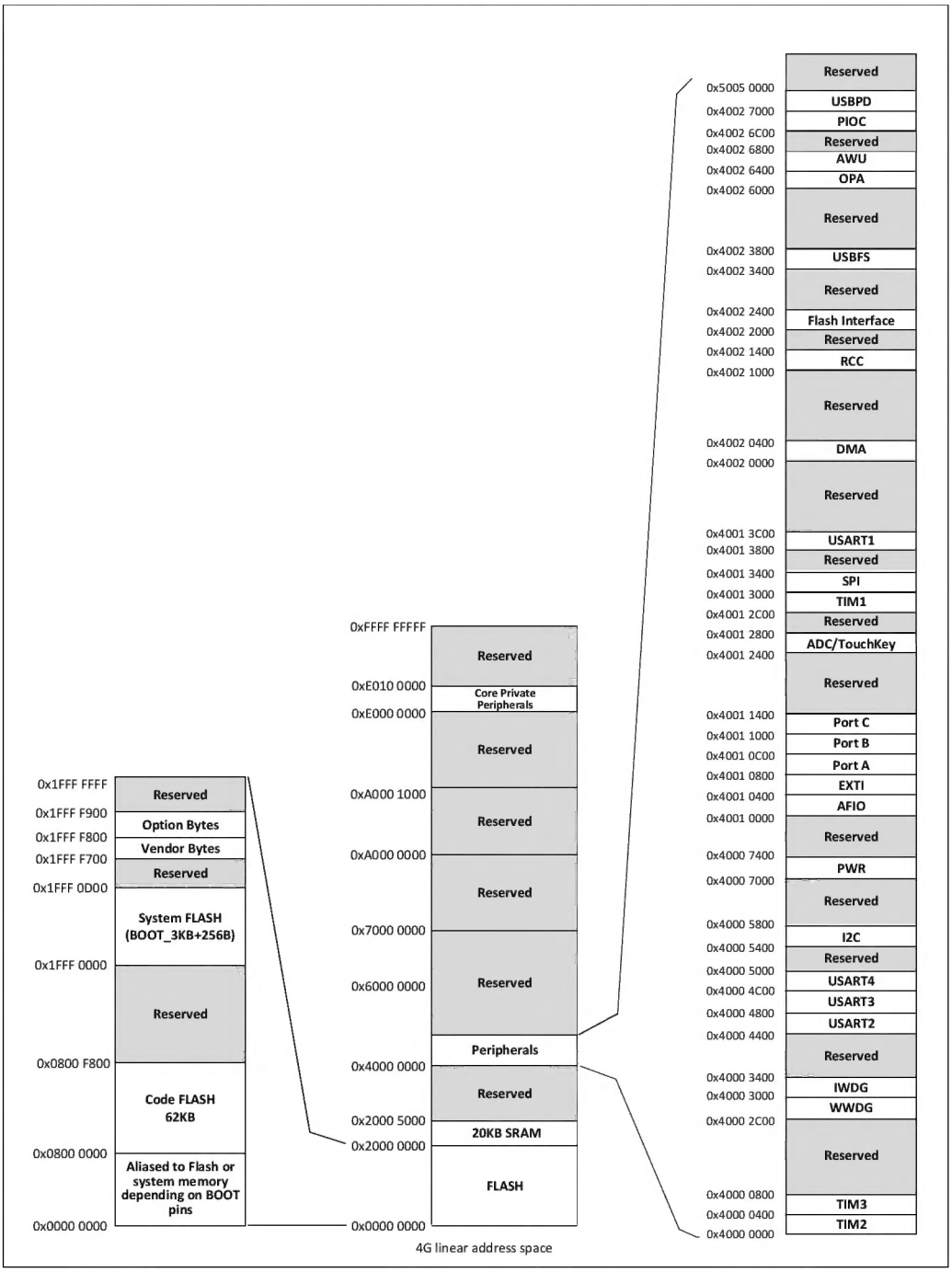
- The data bus (D-Code) connects the core to the FLASH data interface for constant loading and debugging.
- The system bus connects the core to the bus matrix and is used to coordinate accesses to the core, DMA, SRAM and peripherals.
- The DMA bus is responsible for the DMA of the HB master interface connected to the bus matrix, which is accessed by FLASH data, SRAM and peripherals.
- The bus matrix is responsible for the access coordination between the system bus, data bus, DMA bus, SRAM and HB bridge.

## 1.2 Memory Map

The CH32X035 family contains program memory, data memory, core registers, peripheral registers, and more, all addressed in a 4GB linear space.

System storage stores data in small-end format, i.e., low bytes are stored at the low address and high bytes are stored at the high address.

Figure 1-2 Storage image




### 1.2.1 Memory Allocation

Built-in 20KB SRAM, starting address 0x20000000, supports byte, half-word (2 bytes), and full-word (4 bytes)

access.

Built-in up to 62KB program Flash memory (CodeFlash) for storing user applications.

Built-in 3328B System memory (Bootloader) for storing the system bootloader (Factory-cured bootloader).

Built-in 256B space for vendor configuration word storage, factory-cured and unmodifiable by users.

Built-in 256B space for user-selected word storage.

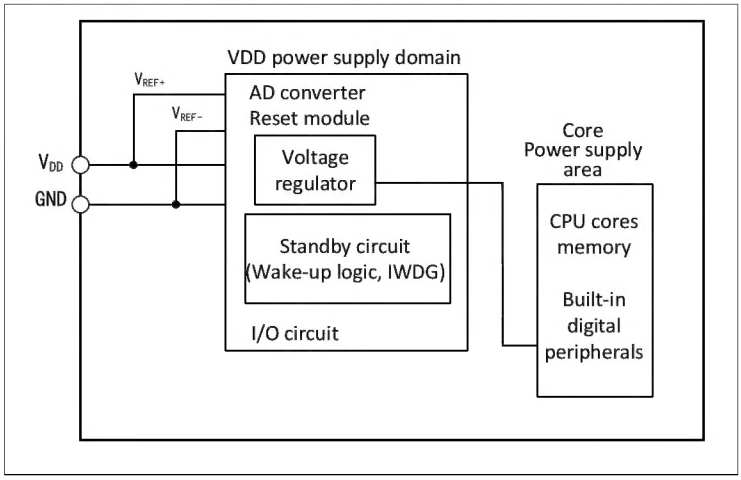
# Chapter 2 Power Control (PWR)

## 2.1 Overview

The system operating voltage $V_{DD}$ ranges from 2 to 5.5V and the built-in voltage regulator provides the low voltage power required by the core.

The $V_{DD}$ and GND pins are dedicated to powering the analogue related circuits in the system, including the ADC etc. $V_{REF+}$ and $V_{REF-}$ are used as reference points for some analogue circuits and are equal to $V_{DD}$ and GND inside the chip.

Figure 2-1 Block diagram of power supply structure

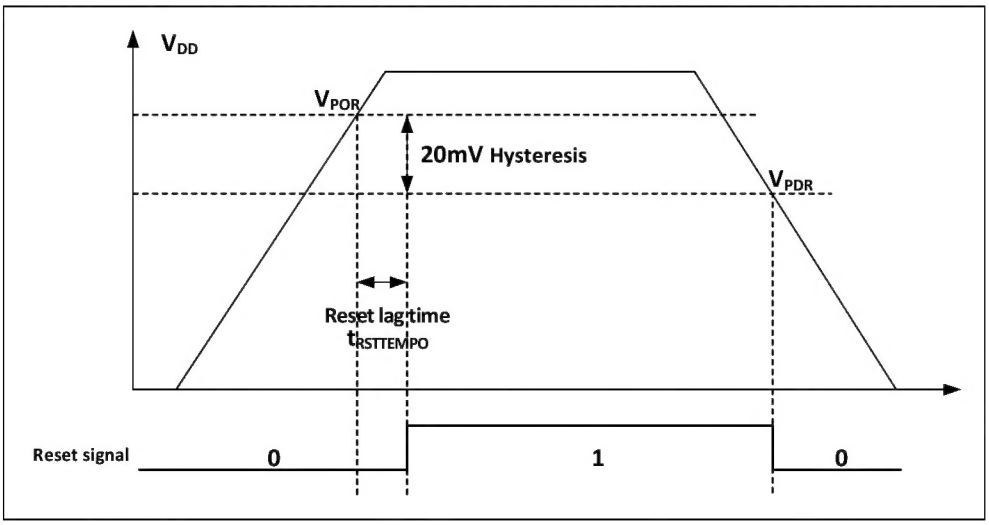



## 2.2 Power Management

### 2.2.1 Power-on Reset and Power-down Reset

The system integrated a power-on reset POR and a power-down reset PDR circuit. When the chip supply voltage $V_{DD}$ falls below the corresponding threshold voltage, the system is reset by the relevant circuit, and no additional external reset circuit is required. Please refer to the corresponding datasheet for the parameters of the power-on threshold voltage $V_{POR}$ and the power-down threshold voltage $V_{PDR}$.

Figure 2-2 Schematic diagram of the operation of POR and PDR



### 2.2.2 Programmable Voltage Detector

The programmable voltage detector, PVD, is mainly used to monitor the change of the main power supply of the system and compare it with the threshold voltage set by PLS[1:0] of the power control register PWR_CTLR, and with the external interrupt register (EXTI) setting, it can generate relevant interrupts to notify the system in time for pre-power down operations such as data saving.

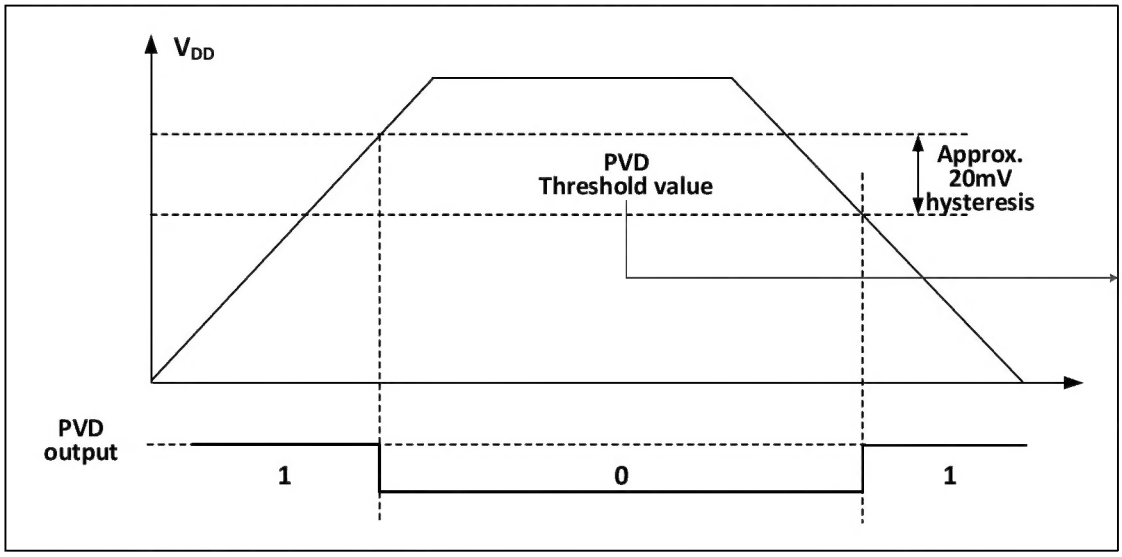
The specific configuration is as follows.

1) Set the PLS[1:0] field of the PWR_CTLR register and select the voltage threshold to be monitored.

2) Optional interrupt handling. the PVD function is internally connected to the rising/falling edge trigger setting of line 26 of the EXTI module, turning on this interrupt (Configuring EXTI) will generate a PVD interrupt when VDD drops below the PVD threshold or rises above the PVD threshold.

3) Read the PVD0 bit of the PWR_CSR status register can obtain the current system mains power in relation to the PLS[1:0] setting threshold and perform the corresponding soft processing.

Figure 2-3 Schematic diagram of PVD operation




## 2.3 Low-power Modes

After a system reset, the microcontroller is in a normal operating state (Run mode), where system power can be saved by reducing the system main frequency or turning off the unused peripheral clock or reducing the operating peripheral clock. If the system does not need to work, you can set the system to enter low-power mode and let the system jump out of this state by specific events.

Microcontrollers currently offer 3 low-power modes, divided in terms of operating differences between processors, peripherals, voltage regulators, etc.

- Sleep mode: The core stops running and all peripherals (Including core private peripherals) are still running.
- Stop mode: Stops all clocks and the system continues to run after waking up.
- Standby mode: Stops all clocks and the system continues to run after waking up.

Table 2-1 Low-power mode list

| Mode | Entry | Wake-up source | Effect on clock | Voltage regulator |
|---|---|---|---|---|
| Sleep | WFI | Any interrupt | Core clock OFF, | Standard |

| | WFE | Wake-up event | no effect on other clocks | mode |
|---|---|---|---|---|
| Stop | Set SLEEPDEEP to 1<br>Clear PDDS to 0<br>WFI or WFE | Any external interrupt/event (EXTI signal), external reset signal on RST, IWDG reset(1) | Disable HIS, peripheral clock | Standard mode |
| Standby | Set SLEEPDEEP to 1<br>Set PDDS to 1<br>WFI or WFE | Any external interrupt/event (EXTI signal), external reset signal on RST, IWDG reset(1) | Disable HIS, peripheral clock | Low-power mode |

*Note: The SLEEPDEEP bit belongs to the core private peripheral control bit, reference PFIC_SCTLR register.*

*(1) Using the IWDG reset function, HSI must not be turned off.*

### 2.3.1 Low-power Configuration Options

- WFI and WFE

WFI: The microcontroller is woken up by an interrupt source with interrupt controller response, and the interrupt service function will be executed first after the system wakes up (Except for microcontroller reset).

WFE: The wakeup event triggers the microcontroller to exit low-power mode. Wake-up events include.

1) Configure an external or internal EXTI line to event mode, when no interrupt controller needs to be configured.
2) Or configure an interrupt source, equivalent to a WFI wakeup, where the system prioritizes the execution of the interrupt service function.
3) Or configure the SEVONPEND bit to turn on peripheral interrupt enable, but not interrupt enable in the interrupt controller, and the interrupt pending bit needs to be cleared after the system wakes up.

- SLEEPONEXIT

  Enable: After executing the WFI or WFE instruction, the microcontroller ensures that all pending interrupt services are exited and then enters low-power mode.

  Not enabled: The microcontroller enters low-power mode immediately after executing the WFI or WFE command.

- SEVONPEND

  Enable: All interrupts or wake-up events can wake up the low-power consumption entered by executing WFE.

  Not enabled: Only interrupts or wake-up events enabled in the interrupt controller can wake up the low-power consumption entered by executing WFE.

### 2.3.2 Sleep Mode

In this mode, all I/O pins keep their state in Run mode and all peripheral clocks are normal, so try to turn off useless peripheral clocks before entering Sleep mode to reduce low-power consumption. This mode takes the shortest time to wake up.

Enter: Configure core register control bit SLEEPDEEP=0, power control register PDDS=0, execute WFI or WFE, optionally SEVONPEND and SLEEPONEXIT.

Exit: Arbitrary interrupt or wakeup event.

### 2.3.3 Stop Mode

Stop mode is a combination of peripheral clock control mechanisms based on the core's deep sleep mode (SLEEPDEEP) and allows the voltage regulator to operate in a much lower power consumption state. In this mode the high frequency clock (HSI) domain is switched off, the SRAM and register contents are maintained and the IO

pin state is held. The system can continue to run after this mode wakes up and the HSI is called the default system clock.

If flash programming is in progress, the system does not enter stop mode until access to memory is complete. Modules that can work in stop mode: Independent Watchdog (IWDG).

Enter: Configure the core register control bit SLEEPDEEP=1, PDDS=0 in the power control register, execute WFI or WFE, optionally SEVONPEND and SLEEPONEXIT.

Exits:

1) Any external interrupt/event (Set in the external interrupt register).

2) External reset, IWDG reset on RST pin.

### 2.3.4 Standby Mode

Standby mode can work modules: Independent Watchdog (IWDG). In this mode, the high frequency clock (HSI) domain is turned off, the SRAM and register contents are retained, and the IO pin states are retained.

Enter: Configure the core register control bit SLEEPDEEP=1, PDDS=1 in the power control register, and execute WFI or WFE, optionally SEVONPEND and SLEEPONEXIT.

Exit:

1) Any interrupt/event (Set in the external interrupt register).

2) AWU event, clock switches to HSI after this wakeup, system does not reset.

### 2.3.5 Auto-wakeup (AWU)

The AWU module enables automatic wake-up without external interrupts. Wake-up from stop or standby mode can be done periodically by programming the time base.

The 47KHz divided clock of the internal high-speed clock HSI is selected as the AWU module clock source.

The AWU module is capable of waking up the MCU from shutdown mode, in order to realize this function, the external interrupt line 27 needs to be configured.

## 2.4 Register Description

Table 2-2 PWR-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_PWR_CTLR | 0x40007000 | Power control register | 0x00000400 |
| R32_PWR_CSR | 0x40007004 | Power control/status register | 0x00000000 |

### 2.4.1 Power Control Register (PWR_CTLR)

Offset address: 0x00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | LP[1:0] | | LP_REG | Reserved | | PLS[1:0] | | Reserved | | | PDDS | Reserved |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | Reserved | RO | Reserved. | 0 |
| [11:10] | LP[1:0] | RW | Software configuration of the FLASH state: | 01b |

| | | | | |
|---|---|---|---|---|
| | | | 00: Idle;<br>x1: Sleep. | |
| 9 | LP_REG | RW | In combination with the LP field, software configuration of FLASH into low-power mode enables:<br>1: The FLASH can be enabled to enter low-power mode;<br>0: The FLASH cannot be enabled into low-power mode by software. | 0 |
| [8:7] | Reserved | RO | Reserved. | 0 |
| [6:5] | PLS[1:0] | RW | PVD voltage monitoring threshold setting.<br>See the Electrical Characteristics section of the datasheet for detailed descriptions.<br>00: 2.12V rising edge/2.1V falling edge;<br>01: 2.32V rising edge / 2.3V falling edge;<br>10: 3.02V rising edge/3V falling edge;<br>11: 4.02V rising edge/4V falling edge. | 0 |
| [4:2] | Reserved | RO | Reserved. | 0 |
| 1 | PDDS | RW | Standby/Stop mode selection bits in power-down deep sleep scenario:<br>1: Enters Standby mode;<br>0: Enters Stop mode. | 0 |
| 0 | Reserved | RO | Reserved. | 0 |

### 2.4.2 Power Control/Status Register (PWR_CSR)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | flash_ack | Reserved | | | | | | PVD0 | Reserved | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:10] | Reserved | RO | Reserved. | 0 |
| 9 | Flash_ack | RO | FLASH status bits:<br>1: Indicates that the FLASH is in low-power mode;<br>0: Indicates that the FLASH is in normal mode. | 0 |
| [8:3] | Reserved | RO | Reserved. | 0 |
| 2 | PVD0 | RO | PVD output status flag bits:<br>1: VDD is below the PVD threshold set by PLS[1:0];<br>0: VDD is above the PVD threshold set by PLS[1:0]. | 0 |
| [1:0] | Reserved | RO | Reserved. | 0 |

# Chapter 3 Reset and Clock Control (RCC)

The controller provides different forms of resets and configurable clock tree structures based on the division of power areas and peripheral power management considerations in the application. This section describes the scope of each clock in the system.

## 3.1 Main Features

- Multiple reset forms
- Multiple clock sources, bus clock management
- Independent management of each peripheral clock: Reset, On, Off
- Supports internal clock output

## 3.2 Reset

The controller provides 2 forms of reset: Power Reset and System Reset.

### 3.2.1 Power Reset

When a power Reset occurs, it will reset all registers.

A power Reset is generated when the following event occurs:

- Power-up/power-down reset (POR/PDR)

### 3.2.2 System Reset

When a system Reset occurs, it will reset the reset flag in addition to the control/status register RCC_RSTSCKR and all the registers. The source of the reset event is identified by looking at the reset status flag bit in the RCC_RSTSCKR register.

A system Reset is generated when one of the following events occurs:

- Low signal on NRST pin (External reset)
- Window watchdog count termination (WWDG reset)
- Independent watchdog count termination (IWDG reset)
- Software reset (SW reset)
- Low-power management reset
- Core deadlock reset
- OPA reset
- USBPD reset
- ADC reset

Window/Independent Watchdog Reset: Generated by the window/independent watchdog peripheral timer count cycle overflow trigger, see its corresponding section for detailed description.

Software reset: Resets the system by resetting the interrupt configuration register PFIC_CFGR to SYSRST position 1 in the programmable interrupt controller PFIC or configuration register PFIC_SCTLR to SYSRST position 1, refer to the corresponding chapter. Low Power Management Reset: A standby mode reset will be enabled by setting STANDY_RST position 0 in the user select byte. At this point after performing the process of entering standby mode, a system reset will be performed instead of entering standby mode. A shutdown mode reset will be enabled by setting STOP_RST to position 0 in the user selection byte. This will perform a system

reset instead of entering shutdown mode after the process of entering shutdown mode has been executed.

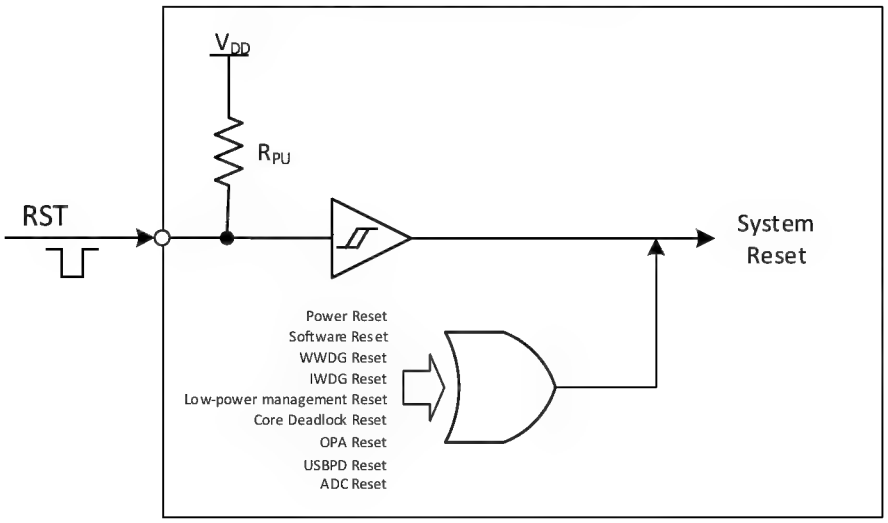
Core Deadlock Reset: With PFIC_SCTLR register RSTEN at 0, a deadlock reset is generated when the core addresses an exception or enters an NMI interrupt, note: A deadlock reset cannot be generated in tuned mode.

OPA reset: When OPA reset enable is on, an OPA reset will be generated when the op-amp output goes high.

USBPD reset: When PD_RST_EN is 1, CH32X035 supports the reset generated by USB PD signal frame Hard Reset; if IE_RX_RESET is also 1, it also supports the reset generated by signal frame Cable Reset. the reset flag generated by USB PD is the same as software reset.

ADC Reset: If the ADC watchdog reset enable is on, an ADC reset is generated when the ADC data is greater than the watchdog high threshold or less than the watchdog low threshold.

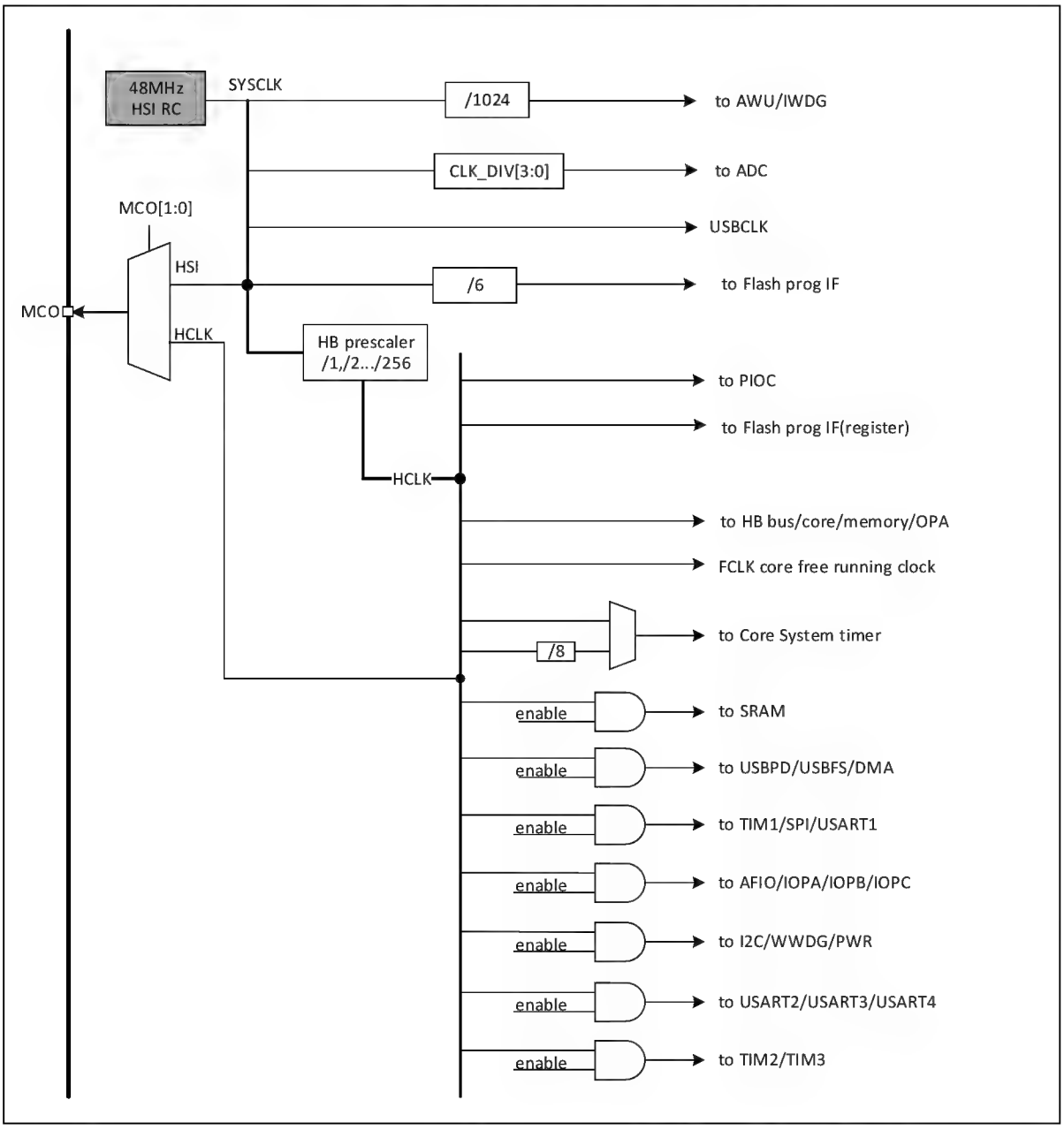
Figure 3-1 System reset structure



# 3.3 Clock

## 3.3.1 System Clock Structure

Figure 3-2 CH32X035 clock tree block diagram



### 3.3.2 High-speed Clock (HSI)

HSI is a high-speed clock signal generated by the system internal 48MHz RC oscillator. HSI RC oscillator can provide system clock without any external devices. It has a short start-up time. HSI is enabled and disabled by setting the HSION bit in the RCC_CTLR register, and the HSIRDY bit indicates whether the HSI RC oscillator is stable or not. The system defaults HSION and HSIRDY to 1 (It is recommended not to turn them off).

- Factory calibration: The difference of manufacturing process will cause different RC oscillation frequency for each chip, so HSI calibration is performed for each chip before it is shipped. After system reset, the factory calibration value is loaded into HSICAL[7:0] of the RCC_CTLR register.
- User tuning: Based on different voltages or ambient temperatures, the application can adjust the HSI frequency by using the HSITRIM[4:0] bits in the RCC_CTLR register.

### 3.3.3 Bus/Peripheral Clock

#### 3.3.3.1 System Clock (SYSCLK)

The default HSI clock is the system clock source.

#### 3.3.3.2 HB Bus Peripheral Clock (HCLK)

The HB bus clocks can be configured by configuring the HPRE[3:0] bits of the RCC_CFGR0 register. The bus clock determines the peripheral interface access clock reference that is mounted below them. Applications can adjust different values to reduce the power consumption when some of the peripherals are operating.

The various bits in the RCC_AHBRSTR, RCC_APB1PRSTR and RCC_APB2PRSTR registers can reset the different peripheral modules to their initial state.

Each bit in the RCC_AHBPCENR, RCC_APB1PCENR, and RCC_APB2PCENR registers can be used to individually turn on or off the communication clock interface for different peripheral modules. When using a peripheral, you first need to turn on its clock enable bit in order to access its registers.

#### 3.3.3.3 Independent Watchdog Clock

If the independent watchdog has been set by hardware configuration or started by software, the HSI oscillator will be forced on and cannot be turned off. After the HSI oscillator is stabilized, the clock is supplied to the IWDG.

#### 3.3.3.4 Microcontroller Clock Output (MCO)

The microcontroller allows outputting clock signals to the MCO pins. The following 2 clock signals can be selected as MCO clock outputs by configuring the alternate push-pull output mode in the corresponding GPIO port registers by configuring the MCO[2:0] bits of the RCC_CFGR0 register.

- System clock (SYSCLK) output
- HSI clock output

## 3.4 Register Description

Table 3-1 RCC-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_RCC_CTLR | 0x40021000 | Clock control register | 0x0000xx83 |
| R32_RCC_CFGR0 | 0x40021004 | Clock configuration register | 0x00000050 |
| R32_RCC_APB2PRSTR | 0x4002100C | PB2 peripheral reset register | 0x00000000 |
| R32_RCC_APB1PRSTR | 0x40021010 | PB1 peripheral reset register | 0x00000000 |
| R32_RCC_AHBPCENR | 0x40021014 | HB peripheral clock enable register | 0x00021004 |

| | | | |
|---|---|---|---|
| R32_RCC_APB2PCENR | 0x40021018 | PB2 peripheral clock enable register | 0x00000000 |
| R32_RCC_APB1PCENR | 0x4002101C | PB1 peripheral clock enable register | 0x00000000 |
| R32_RCC_RSTSCKR | 0x40021024 | Control/status register | 0x0C000000 |
| R32_RCC_AHBRSTR | 0x40021028 | HB peripheral reset register | 0x00000000 |

### 3.4.1 Clock Control Register (RCC_CTLR)

Offset address: 0x00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| HSICAL[7:0] | | | | | | | | HSITRIM[4:0] | | | | | Reserved | HSIRDY | HSION |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RW | Reserved. | 0 |
| [15:8] | HSICAL[7:0] | RO | Internal high-speed clock calibration values, which are automatically initialized at system startup. | X |
| [7:3] | HSITRIM[4:0] | RW | Internal high-speed clock adjustment value.<br>The user can enter an adjustment value to superimpose on the HSICAL[7:0] value to adjust the frequency of the internal HSI RC oscillator based on voltage and temperature variations.<br>The default value is 16, which can adjust the HSI to 48MHz ±1%; the change of HSICAL is adjusted about 110KHz per step. | 10000b |
| 2 | Reserved | RO | Reserved. | 0 |
| 1 | HSIRDY | RO | Internal high-speed clock stable ready flag bit (Set by hardware).<br>1: The internal high-speed clock is stable;<br>0: The internal high-speed clock is not stable.<br>*Note: After the HSION bit is cleared to 0, it takes 6 HSI cycles for the bit to be cleared to 0.* | 1 |
| 0 | HSION | RW | Internal high-speed clock enable control bit:<br>1: Enable the HSI oscillator.<br>0: Disable the HSI oscillator. | 1 |

### 3.4.2 Clock Configuration Register0 (RCC_CFGR0)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | MCO[2:0] | | | Reserved | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | HPRE[3:0] | | | | Reserved | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:27] | Reserved | RO | Reserved. | 0 |
| [26:24] | MCO[2:0] | RW | Microcontroller MCO pin clock output control:<br>100: System clock (SYSCLK) output;<br>101: HIS clock output;<br>Other: No clock output. | 0 |

| | | | | |
|---|---|---|---|---|
| [23:8] | Reserved | RO | Reserved. | 0 |
| [7:4] | HPRE[3:0] | RW | HB clock source prescaler control:<br>0000: Prescaler off.<br>0001: SYSCLK divided by 2.<br>0010: SYSCLK divided by 3.<br>0011: SYSCLK divided by 4.<br>0100: SYSCLK divided by 5.<br>0101: SYSCLK divided by 6.<br>0110: SYSCLK divided by 7.<br>0111: SYSCLK divided by 8.<br>1000: SYSCLK divided by 2.<br>1001: SYSCLK divided by 4.<br>1010: SYSCLK divided by 8.<br>1011: SYSCLK divided by 16.<br>1100: SYSCLK divided by 32.<br>1101: SYSCLK divided by 64.<br>1110: SYSCLK divided by 128.<br>1111: SYSCLK divided by 256.<br>*Note: When the prescaler factor of the HB clock source is greater than 1, the prefetch buffer must be turned on.* | 0101b |
| [3:0] | Reserved | RO | Reserved. | 0 |

### 3.4.3 PB2 Peripheral Reset Register (RCC_APB2PRSTR)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | USART1 RST | Reserved | SPI1 RST | TIM1 RST | Reserved | ADC1 RST | Reserved | | | | IOPC RST | IOPB RST | IOPA RST | Reserved | AFIO RST |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| 14 | USART1RST | RW | USART1 interface reset control:<br>1: Reset module; 0: No effect. | 0 |
| 13 | Reserved | RO | Reserved. | 0 |
| 12 | SPI1RST | RW | SPI1 interface reset control:<br>1: Reset module; 0: No effect. | 0 |
| 11 | TIM1RST | RW | TIM1 module reset control.<br>1: Reset module; 0: No effect. | 0 |
| 10 | Reserved | RO | Reserved | 0 |
| 9 | ADC1RST | RW | ADC1 module reset control:<br>1: Reset module; 0: No effect. | 0 |
| [8:5] | Reserved | RO | Reserved. | 0 |
| 4 | IOPCRST | RW | PC port module reset control for I/O:<br>1: Reset module; 0: No effect. | 0 |
| 3 | IOPBRST | RW | PB port module reset control for I/O:<br>1: Reset module; 0: No effect. | 0 |
| 2 | IOPARST | RW | PA port module reset control for I/O:<br>1: Reset module; 0: No effect. | 0 |
| 1 | Reserved | RO | Reserved. | 0 |
| 0 | AFIORST | RW | I/O auxiliary function module reset control:<br>1: Reset module; 0: No effect. | 0 |

### 3.4.4 PB1 Peripheral Reset Register (RCC_APB1PRSTR)

Offset address: 0x10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | PWRRST | Reserved | | | | | | I2C1RST | Reserved | USART4RST | USART3RST | USART2RST | Reserved |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | WWDGRST | Reserved | | | | | | | | | TIM3RST | TIM2RST |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:29] | Reserved | RO | Reserved. | 0 |
| 28 | PWRRST | RW | Power interface module reset control:<br>1: Reset module; 0: No effect. | 0 |
| [27:22] | Reserved | RO | Reserved. | 0 |
| 21 | I2C1RST | RW | I2C1 interface reset control:<br>1: Reset module; 0: No effect. | 0 |
| 20 | Reserved | RW | Reserved. | 0 |
| 19 | USART4RST | RW | USART4 interface reset control:<br>1: Reset module; 0: No effect. | 0 |
| 18 | USART3RST | RW | USART3 interface reset control:<br>1: Reset module; 0: No effect. | 0 |
| 17 | USART2RST | RW | USART2 interface reset control:<br>1: Reset module; 0: No effect. | 0 |
| [16:12] | Reserved | RO | Reserved. | 0 |
| 11 | WWDGRST | RW | Window watchdog reset control:<br>1: Reset module; 0: No effect. | 0 |
| [10:2] | Reserved | RO | Reserved. | 0 |
| 1 | TIM3RST | RW | Timer 3 module reset control:<br>1: Reset module; 0: No effect. | 0 |
| 0 | TIM2RST | RW | Timer 2 module reset control:<br>1: Reset module; 0: No effect. | 0 |

### 3.4.5 HB Peripheral Clock Enable Register (RCC_AHBPCENR)

Offset address: 0x14

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | USBPD | Reserved |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | USBFSEN | Reserved | | | | | | | | | SRAMEN | Reserved | DMA1EN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:18] | Reserved | RO | Reserved. | 0 |
| 17 | USBPD | RW | USBPD clock enable:<br>1： USBPD clock on; | 1 |

| | | | | |
|---|---|---|---|---|
| | | | 0：USBPD clock off. | |
| [16:13] | Reserved | RO | Reserved. | 0 |
| 12 | USBFSEN | RW | USBFS module clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 1 |
| [11:3] | Reserved | RO | Reserved. | 0 |
| 2 | SRAMEN | RW | SRAM interface module clock enable bit:<br>1: SRAM interface module clock on in Sleep mode;<br>0: SRAM interface module clock off in Sleep mode. | 1 |
| 1 | Reserved | RO | Reserved. | 0 |
| 0 | DMA1EN | RW | DMA1 module clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |

### 3.4.6 PB2 Peripheral Clock Enable Register (RCC_APB2PCENR)

Offset address: 0x18

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | USART1EN | Reserved | SPI1EN | TIM1EN | Reserved | ADC1EN | Reserved | | | | IOPCEN | IOPBEN | IOPAEN | Reserved | AFIOEN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| 14 | USART1EN | RW | USART1 interface clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 13 | Reserved | RO | Reserved. | 0 |
| 12 | SPI1EN | RW | SPI1 interface clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 11 | TIM1EN | RW | TIM1 module clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 10 | Reserved | RO | Reserved. | 0 |
| 9 | ADC1EN | RW | ADC1 module clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| [8:5] | Reserved | RO | Reserved. | 0 |
| 4 | IOPCEN | RW | PC port module clock enable bit for I/O:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 3 | IOPBEN | RW | PB port module clock enable bit for I/O:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 2 | IOPAEN | RW | PA port module clock enable bit for I/O:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 1 | Reserved | RO | Reserved. | 0 |
| 0 | AFIOEN | RW | I/O auxiliary function module clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |

### 3.4.7 PB1 Peripheral Clock Enable Register (RCC_APB1PCENR)

Offset address: 0x1C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | PWREN | Reserved | | | | | | I2C1EN | Reserved | USART4EN | USART3EN | USART2EN | Reserved |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

| Reserved | WWDGEN | Reserved | TIM3EN | TIM2EN |
|---|---|---|---|---|

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:29] | Reserved | RO | Reserved. | 0 |
| 28 | PWREN | RW | Power interface module clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| [27:22] | Reserved | RO | Reserved. | 0 |
| 21 | I2C1EN | RW | I2C1 interface clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 20 | Reserved | RO | Reserved. | 0 |
| 19 | USART4EN | RW | USART4 interface clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 18 | USART3EN | RW | USART3 interface clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 17 | USART2EN | RW | USART2 interface clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| [16:12] | Reserved | RO | Reserved. | 0 |
| 11 | WWDGEN | RW | Window watchdog clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| [10:2] | Reserved | RO | Reserved. | 0 |
| 1 | TIM3EN | RW | Timer 3 module clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |
| 0 | TIM2EN | RW | Timer 2 module clock enable bit:<br>1: Module clock is on; 0: Module clock is off. | 0 |

### 3.4.8 Control/Status Register (RCC_RSTSCKR)

Offset address: 0x24

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LPWRRSTF | WWDGRSTF | IWDGRSTF | SFTRSTF | PORRSTF | PINRSTF | OPARSTF | RMVF | Reserved | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 31 | LPWRRSTF | RW | Low-power reset flag:<br>1: Occurrence of low-power resets.<br>0: No low-power reset occurs.<br>Set to 1 by hardware when a low-power management reset occurs; cleared by software writing of the RMVF bit. | 0 |
| 30 | WWDGRSTF | RW | Window watchdog reset flag:<br>1: Occurrence of a window watchdog reset.<br>0: No window watchdog reset occurs.<br>Set to 1 by hardware when a window watchdog reset occurs; cleared by software writing of the RMVF bit. | 0 |
| 29 | IWDGRSTF | RW | Independent watchdog reset flag:<br>1: Occurrence of an independent watchdog reset.<br>0: No independent watchdog reset occurs.<br>Set to 1 by hardware when an independent watchdog | 0 |

| | | | | |
|---|---|---|---|---|
| | | | reset occurs; cleared by software writing of the RMVF bit. | |
| 28 | SFTRSTF | RW | Software reset flag:<br>1: Software reset occurs.<br>0: No software reset occurs.<br>Set to 1 by hardware when a software reset occurs; software write RMVF bit cleared. | 0 |
| 27 | PORRSTF | RW | Power-up/power-down reset flag:<br>1: Power-up/power-down reset occurs.<br>0: No power-up/power-down reset occurs.<br>Set to 1 by hardware when power-up/power-down reset occurs; cleared by software writing of RMVF bit. | 1 |
| 26 | PINRSTF | RW | External manual reset (NRST pin) flag:<br>1: Occurrence of NRST pin reset.<br>0: No NRST pin reset occurs.<br>Set to 1 by hardware when NRST pin reset occurs; cleared by software writing of RMVF bit. | 0 |
| 25 | OPARSTF | RW | OPA reset flag control:<br>1: OPA reset flag;<br>0: No effect. | 0 |
| 24 | RMVF | RW | Clear reset flag control:<br>1: Clear the reset flag.<br>0: No effect. | 0 |
| [23:0] | Reserved | RO | Reserved. | 0 |

*Note: Except for the reset flag which can only be cleared by power-on reset, others are cleared by system Reset.*

### 3.4.9 HB Peripheral Reset Register (RCC_AHBRSTR)

Offset address: 0x28

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | USBPD RST | Reserved |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | PIOC RST | USBF SRST | Reserved | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:18] | Reserved | RO | Reserved. | 0 |
| 17 | USBPDRST | RW | USBPD reset control:<br>1: Reset module; 0: No effect. | 0 |
| [16:14] | Reserved | RO | Reserved. | 0 |
| 13 | PIOCRST | RW | PIOC reset control:<br>1: Reset module; 0: No effect. | 0 |
| 12 | USBFSRST | RW | USBFS module reset control:<br>1: Reset module; 0: No effect. | 0 |
| [11:0] | Reserved | RO | Reserved. | 0 |

# Chapter 4 Auto-wakeup (AWU)

The AWU module enables automatic wake-up without external interrupts. Wake-up from stop or standby mode can be done periodically by programming the time base.

## 4.1 Main Features

- 6-bit self-adding counter
- Select the 47KHz divided clock of the internal high-speed clock HSI as the AWU module clock source, which can be operated in low-power mode
- Wake-up condition: when the counter counts to a value equal to the one written in

## 4.2 Function Description

The AWU module is capable of waking up the MCU from low power mode. To achieve this function, the external interrupt line 27 needs to be configured as a rising edge interrupt.

## 4.3 Register Description

Table 4-1 AWU-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_AWU_CSR | 0x40026400 | Control/Status register | 0x0000 |
| R16_AWU_WR | 0x40026404 | Wake-up window register | 0x003f |
| R16_AWU_PSC | 0x40026408 | Prescaler register | 0x0000 |

### 4.3.1 AWU Control/Status Register (AWU_CSR)

Offset address: 0x00

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | AWUEN | Reserved |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:2] | Reserved | RO | Reserved. | 0 |
| 1 | AWUEN | RW | AWU wakeup enable:<br>1: AWU on;<br>0: AWU off. | 0 |
| 0 | Reserved | RO | Reserved. | 0 |

### 4.3.2 AWU Wake-up Window Register (AWU_WR)

Offset address: 0x04

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | AWU_APR[5:0] | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| [15:6] | Reserved | RO | Reserved. | 0 |
| [5:0] | AWU_APR[5:0] | RW | The window value, which is used to compare with the counter value, is equal to produce a wake-up signal. | 0x3F |

### 4.3.3 AWU Prescaler Register (AWU_PSC)

Offset address: 0x08

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | AWU_TBR[3:0] | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:4] | Reserved | RO | Reserved. | 0 |
| [3:0] | AWU_TBR[3:0] | RW | Frequency division factor:<br>0000: Prescaler off;<br>0001: Prescaler off;<br>0010: Divided by 2;<br>0011: Divided by 4;<br>0100: Divided by 8;<br>0101: Divided by 16;<br>0110: Divided by 32;<br>0111: Divided by 64;<br>1000: Divided by 128;<br>1001: Divided by 256;<br>1010: Divided by 512;<br>1011: Divided by 1024;<br>1100: Divided by 2048;<br>1101: Divided by 4096;<br>1110: Divided by 10240;<br>1111: Divided by 61440. | 0 |

# Chapter 5 Independent Watchdog (IWDG)

The system is equipped with an independent watchdog (IWDG) to detect logic errors and software faults caused by external environmental disturbances. the IWDG clock source is derived from the LSI and can run independently of the main program, making it suitable for applications requiring low accuracy.

## 5.1 Main Features

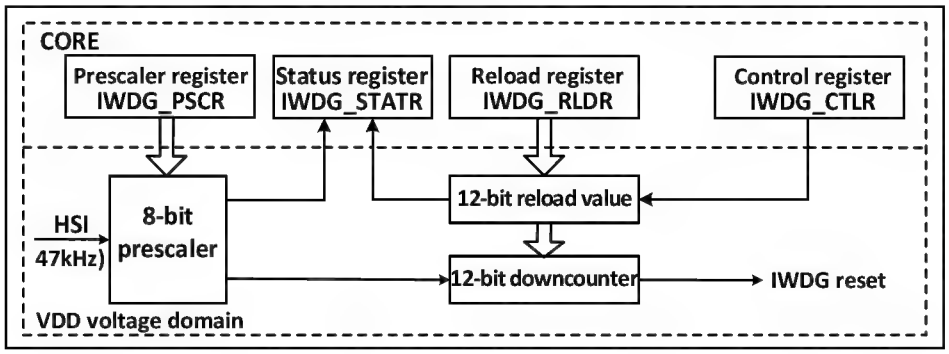
- 12-bit self-subtracting counter
- Clock source HSI divided by 1024 (47KHz), can run in low-power mode
- Reset condition: Counter value is reduced to 0

## 5.2 Function Description

### 5.2.1 Principle and Application

The independent watchdog is clocked from the HSI clock and its function remains functional during shutdown and Standby modes. When the watchdog counter self-decreases to 0, a system Reset will be generated, so the timeout is (Reload value + 1) clock.

Figure 5-1 Block diagram of the structure of the independent watchdog




- Enable independent watchdog

After a system reset, the watchdog is off and writing 0xCCCC to the IWDG_CTLR register turns the watchdog on, after which it cannot be turned off again unless a reset occurs.

If the hardware independent watchdog enable bit (IWDG_SW) is turned on at the user-selected word, IWDG will be fixed on after a microcontroller reset.

- Watchdog configuration

The watchdog is internally a 12-bit counter that runs decreasingly. When the counter value decreases to 0, a system Reset will occur. To turn on the IWDG function, the following actions need to be performed.

1) Counting time base: IWDG clock source HIS divided by 1024, set the HSI crossover value clock as the counting time base of IWDG through the IWDG_PSCR register. The operation method first writes 0x5555 to the IWDG_CTLR register, and then modifies the crossover value in the IWDG_PSCR register. the PVU bit in the IWDG_STATR register indicates the update status of the crossover value, and the crossover value can be modified and read out only when the update is completed.
2) Reload value: Used to update the current value of the counter in the standalone watchdog and the counter is

decremented by this value. The RVU bit in the IWDG_STATR register indicates the update status of the reload value, and the IWDG_RLDR register can be modified and read out only when the update is completed.

3) Watchdog enable: write 0xCCCC to the IWDG_CTLR register to enable the watchdog function.
4) Feed the dog: i.e., flush the current counter value before the watchdog counter decrements to 0 to prevent a system reset from occurring. Write 0xAAAA to the IWDG_CTLR register to allow the hardware to update the IWDG_RLDR register value to the watchdog counter. This action needs to be executed regularly after the watchdog function is turned on, otherwise a watchdog reset action will occur.

### 5.2.2 Debug Mode

When the system enters Debug mode, the counter of IWDG can be configured by the debug module register to continue or stop.

## 5.3 Register Description

Table 5-1 IWDG-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_IWDG_CTLR | 0x40003000 | Control register | 0x0000 |
| R16_IWDG_PSCR | 0x40003004 | Prescaler register | 0x0000 |
| R16_IWDG_RLDR | 0x40003008 | Reload register | 0x0FFF |
| R16_IWDG_STATR | 0x4000300C | Status register | 0x0000 |

### 5.3.1 Control Register (IWDG_CTLR)

Offset address: 0x00

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KEY[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | KEY | WO | Operate the key value lock.<br>00xAAAA: Feed the dog. Loading of the IWDG_RLDR register value into the independent watchdog counter.<br>0x5555: Allows modification of the IWDG_PSCR and IWDG_ RLDR registers.<br>0xCCCC: Start the watchdog, which is not restricted if the hardware watchdog is enabled (user-selected word configuration). | 0 |

### 5.3.2 Prescaler Register (IWDG_PSCR)

Offset address: 0x04

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | PR[2:0] | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:3] | Reserved | RO | Reserved | 0 |
| [2:0] | PR[2:0] | RW | IWDG clock division factor, write 0x5555 to KEY before modifying this field.<br>000: Divided by 4; 001: Divided by 8. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 010: Divided by 16; 011: Divided by 32.<br>100: Divided by 64; 101: Divided by 128.<br>110: Divided by 256; 111: Divided by 256.<br>IWDG counting time base = HSI/1024/divide factor.<br>*Note: Before reading the value of this field, make sure the PVU bit in the IWDG_STATR register is 0, otherwise the read value is invalid.* | |

### 5.3.3 Reload Register (IWDG_RLDR)

Offset address: 0x08

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | RL[11:0] | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:12] | Reserved | RO | Reserved | 0 |
| [11:0] | RL[11:0] | RW | Counter reload value. Write 0x5555 to the KEY before modifying this field.<br>When 0xAAAA is written to the KEY, the value of this field will be loaded into the counter by hardware, and the counter will then count decreasingly from this value.<br>*Note: Before reading or writing the value of this field, make sure the RVU bit in the IWDG_STATR register is 0, otherwise reading or writing this field is invalid.* | 0xFFF |

### 5.3.4 Status Register (IWDG_STATR)

Offset address: 0x0C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | RVU | PVU |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:2] | Reserved | RO | Reserved | 0 |
| 1 | RVU | RO | Reload value update flag bit. Hardware set or clear 0.<br>1: Reload value update is in progress.<br>0: End of reload update (Up to 5 HSI cycles).<br>*Note: The reload value register IWDG_RLDR can only be accessed read or write after the RVU bit is cleared to 0.* | 0 |
| 0 | PVU | RO | Clock division factor update flag bit. Hardware set or clear 0.<br>1: Clock division value update is in progress.<br>0: End of clock division value update (Up to 5 HSI cycles).<br>*Note: The crossover factor register IWDG_PSCR can only be accessed read or write after the PVU bit is cleared to 0.* | 0 |

*Note: After the prescaler or reload value is updated, it is not necessary to wait for the RVU or PVU to reset, and the following code can continue to be executed. (This write operation will continue to be executed to completion*

*even in low-power mode.)*

# Chapter 6 Window Watchdog (WWDG)

Window Watchdog is generally used to monitor system operation for software faults such as external disturbances, unforeseen logic errors, and other conditions. It requires a counter refresh (dog feeding) within a specific window time (With upper and lower limits), otherwise earlier or later than this window time the watchdog circuit will generate a system Reset.

## 6.1 Main Features

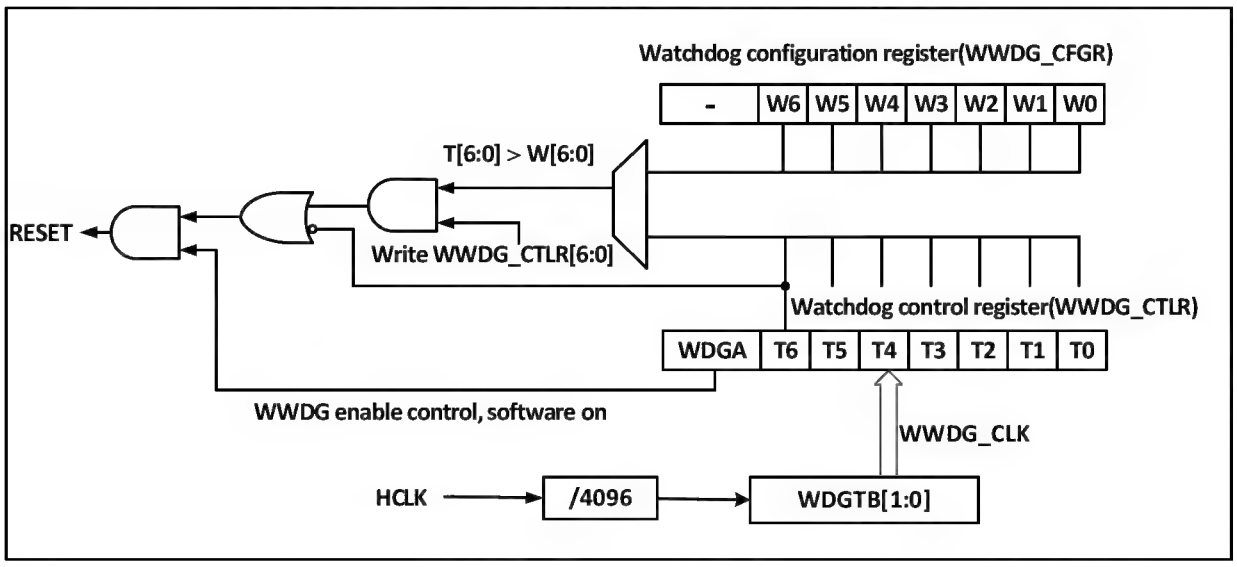
- Programmable 7-bit downcounter
- Biconditional reset: the downcounter value is less than 0x40, or the counter value is reloaded outside the window time
- Wake Up Early Notification (EWI) function for timely dog feeding action to prevent system Reset

## 6.2 Function Description

### 6.2.1 Principle and Application

The window watchdog operation is based on a 7-bit downcounter, which is mounted under the HB bus and counts the dividing frequency of the time base WWDG_CLK source (HCLK/4096) clock with the dividing factor set in the WDGTB[1:0] field in the configuration register WWDG_CFGR. The downcounter is in the free-running state, and the counter keeps cycling downcount regardless of whether the watchdog function is on or not. As shown in Figure 6-1, the block diagram of the internal structure of the window watchdog.

Figure 6-1 Block diagram of Window Watchdog structure




- Enable Window Watchdog

After a system Reset, the watchdog is off. Setting the WDGA bit of the WWDG_CTLR register enables the watchdog, and then it cannot be turned off again unless a reset occurs.

*Note: The watchdog function can be stopped indirectly by setting the RCC_APB1PCENR register to turn off the clock source of WWDG and suspend the WWDG_CLK count, or by setting the RCC_APB1PRSTR register to reset the WWDG module, which is equivalent to the role of reset.*

- Watchdog Configuration

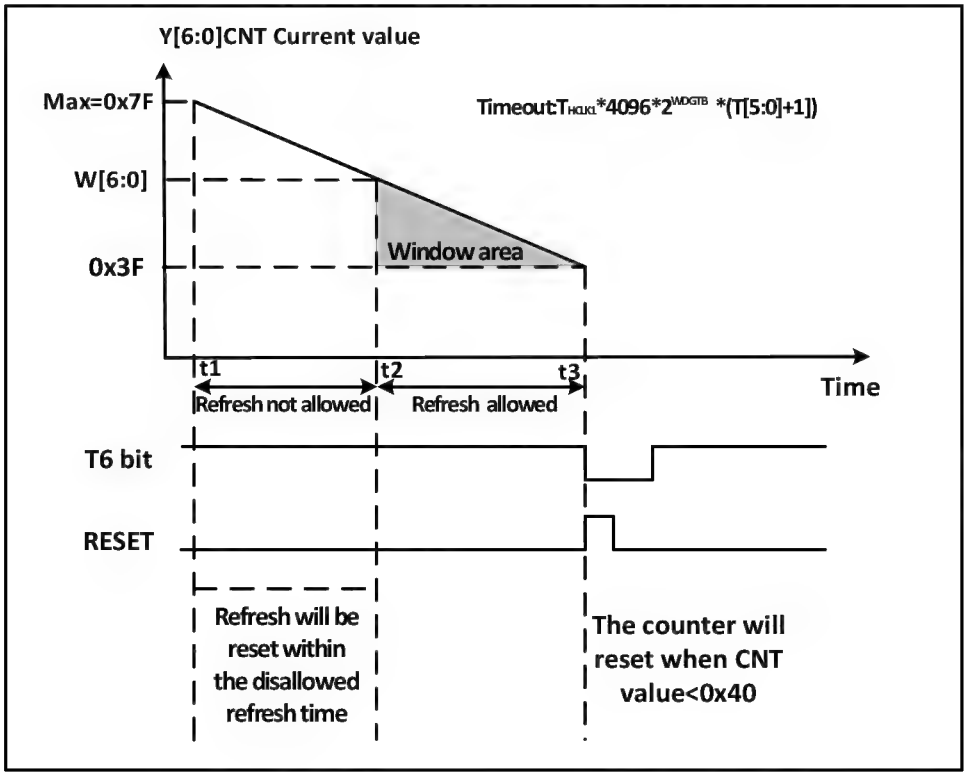
The watchdog is internally a 7-bit counter that runs in a continuous decreasing cycle and supports read and write access. To use the watchdog reset function, the following actions need to be performed.

1) Counting time base: via the WDGTB[1:0] bit field of the WWDG_CFGR register, note that the WWDG module clock of the RCC unit should be turned on.
2) Window counter: Set the W[6:0] bit field of WWDG_CFGR register, this counter is used by hardware as a comparison with the current counter, the value is configured by user software and will not change. It is used as the upper limit value of the window time.
3) Watchdog enable: WDG_CTLR register WDGA bit software set to 1, to turn on the watchdog function, you can system reset.
4) Feed the dog: i.e., refresh the current counter value and configure the T[6:0] bit field of the WWDG_CTLR register. This action needs to be executed within the periodic window time after the watchdog function is turned on, otherwise a watchdog reset action will occur.

- Dog feeding window time

As shown in Figure 6-2, the gray area is the monitoring window area of the window watchdog, whose upper time t2 corresponds to the point in time when the current counter value reaches the window value W[6:0]; its lower time t3 corresponds to the point in time when the current counter value reaches 0x3F. This area time t2<t<t3 can be fed with a dog operation (Write T[6:0]) to refresh the current counter value.

- Figure 6-2 Counting mode of Window Watchdog




- Watchdog reset

1) When the value of T[6:0] counter changes from 0x40 to 0x3F due to no timely dog feeding operation, a "window watchdog reset" will occur and a system reset will be generated. That is, the T6-bit is detected as 0 by the hardware and a system reset will occur.

*Note: The application can write T6-bit to 0 by software to achieve system Reset, which is equivalent to software reset function.*

2) When the counter refresh action is executed within the disallowed dog feeding time, i.e., the write T[6:0] bit field is operated within $t_1 \leqslant t \leqslant t_2$ time, a "window watchdog reset" will occur and a system Reset will be generated.

- Wake up in advance

To prevent the system Reset caused by not refreshing the counter in time, the watchdog module provides an early wakeup interrupt (EWI) notification. When the counter self-decreases to 0x40, an early wake-up signal is generated and the EWIF flag is set to 1. If the EWI bit is set, a window watchdog interrupt will be triggered at the same time. At this time, there is 1 counter clock cycle (Self-decrement to 0x3F) before the hardware reset, and the application can perform the dog feeding operation instantly within this time.

### 6.2.2 Debug Mode

When the system enters Debug mode, the counter of WWDG can be configured by the debug module register to continue or stop.

## 6.3 Register Description

Table 6-1 WWDG-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_WWDG_CTLR | 0x40002C00 | Control register | 0x007F |
| R16_WWDG_CFGR | 0x40002C04 | Configuration Register | 0x007F |
| R16_WWDG_STATR | 0x40002C08 | Status Register | 0x0000 |

### 6.3.1 Control Register (WWDG_CTLR)

Offset address: 0x00

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | WDGA | T[6:0] | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved | 0 |
| 7 | WDGA | RW1 | Window watchdog reset enable bit.<br>1: Turn on the watchdog function (Which generates a reset signal).<br>0: Disable the watchdog function. Software write 1 is on, but only allows hardware to clear 0 after reset. | 0 |
| [6:0] | T[6:0] | RW | The 7-bit self-decrement counter decrements by 1 every $4096*2^{WDGTB}$ HCLK cycles. A watchdog reset is generated when the counter decrements from 0x40 to 0x3F, i.e., when T6 jumps to 0. | 0x7F |

### 6.3.2 Configuration Register (WWDG_CFGR)

Offset address: 0x04

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | EWI | WDGTB[1:0] | | W[6:0] | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:10] | Reserved | RO | Reserved | 0 |
| 9 | EWI | RW1 | Early wakeup interrupt enable bit.<br>If this position is 1, an interrupt is generated when the counter value reaches 0x40. This bit can only be invited to 0 by hardware after a reset. | 0 |
| [8:7] | WDGTB[1:0] | RW | Window watchdog clock division selection.<br>00: Divided by 1, counting time base = HCLK/4096.<br>01: Divided by 2, counting time base = HCLK /4096/2.<br>10: Divided by 4, counting time base = HCLK /4096/4.<br>11: Divided by 8, counting time base = HCLK /4096/8. | 0 |
| [6:0] | W[6:0] | RW | Window watchdog 7-bit window value. Used to compare with the counter value. The feed dog operation can only be performed when the counter value is less than the window value and greater than 0x3F. | 0x7F |

### 6.3.3 Status Register (WWDG_STATR)

Offset address: 0x08

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | EWIF |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:1] | Reserved | WO | Reserved | 0 |
| 0 | EWIF | RW0 | Wake up the interrupt flag bit early.<br>When the counter reaches 0x40, this bit is set in hardware and must be cleared to 0 by software; the user setting is invalid. Even if the EWI is not set, this bit will still be set as usual when the event occurs. | 0 |

# Chapter 7 Interrupt and Events (PFIC)

The CH32X035 series has a built-in Programmable Fast Interrupt Controller (PFIC) that supports up to 255 interrupt vectors. The current system manages 39 peripheral interrupt channels and 7 core interrupt channels, the others are reserved.

## 7.1 Main Features

### 7.1.1 PFIC

- 39 peripheral interrupts, each interrupt request has independent trigger and mask control bits, with dedicated status bits
- Programmable multi-level interrupt nesting, maximum nesting depth 2 levels, hardware stack depth 2 levels
- Fast interrupt entry and exit mechanism, hardware automatic stacking and recovery
- Vector Table Free (VTF) interrupt response mechanism, 4-way programmable direct access to interrupt vector addresses

## 7.2 System Timer

- CH32X035 Series

The core comes with a 64-bit add counter (SysTick) that supports HCLK or HCLK/8 as a time base with high priority and can be used as a time reference after calibration.

## 7.3 Vector Table of Interrupts and Exceptions

Table 7-1 CH32X035 series vector table

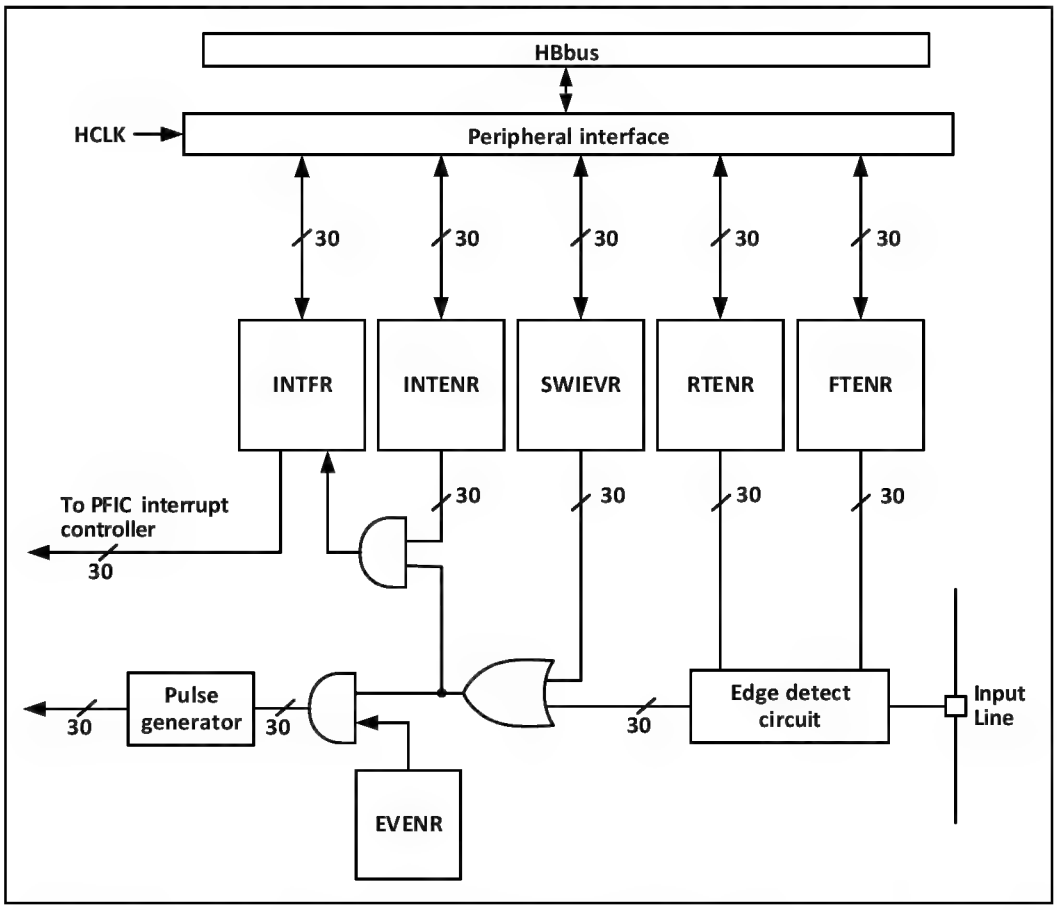
| No. | Priority | Type | Name | Description | Entrance address |
|---|---|---|---|---|---|
| 0 | - | - | - | - | 0x00000000 |
| 1 | - | - | - | - | 0x00000004 |
| 2 | -5 | fixed | NMI | Non-maskable interrupts | 0x00000008 |
| 3 | -4 | fixed | HardFault | Abnormal interruptions | 0x0000000C |
| 4 | - | - | - | Reserved | 0x00000010 |
| 5 | -3 | fixed | Ecall-M | Machine mode callback interrupt | 0x00000014 |
| 6-7 | - | - | - | Reserved | 0x00000018-0x0000001C |
| 8 | -2 | fixed | Ecall-U | User mode callback interrupt | 0x00000020 |
| 9 | -1 | fixed | BreakPoint | Brake callback interruption | 0x00000024 |
| 10-11 | - | - | - | Reserved | 0x00000028-0x0000002C |
| 12 | 0 | programmable | SysTick | System timer interrupt | 0x00000030 |
| 13 | - | - | - | Reserved | 0x00000034 |
| 14 | 1 | programmable | SW | Software interrupt | 0x00000038 |
| 15 | - | - | - | Reserved | 0x0000003C |
| 16 | 2 | programmable | WWDG | Window timer interrupt | 0x00000040 |
| 17 | 3 | programmable | PVD | Supply voltage detection interrupt (EXTI) | 0x00000044 |
| 18 | 4 | programmable | FLASH | FLASH interrupt | 0x00000048 |
| 19 | - | - | - | Reserved | 0x0000004C |

| | | | | | |
|---|---|---|---|---|---|
| 20 | 6 | programmable | EXTI7_0 | EXTI7_0 interrupt | 0x00000050 |
| 21 | 7 | programmable | AWU | AWU interrupt | 0x00000054 |
| 22 | 8 | programmable | DMA1_CH1 | DMA1CH1 interrupt | 0x00000058 |
| 23 | 9 | programmable | DMA1_CH2 | DMA1CH2 interrupt | 0x0000005C |
| 24 | 10 | programmable | DMA1_CH3 | DMA1CH3 interrupt | 0x00000060 |
| 25 | 11 | programmable | DMA1_CH4 | DMA1CH4 interrupt | 0x00000064 |
| 26 | 12 | programmable | DMA1_CH5 | DMA1CH5 interrupt | 0x00000068 |
| 27 | 13 | programmable | DMA1_CH6 | DMA1CH6 interrupt | 0x0000006C |
| 28 | 14 | programmable | DMA1_CH7 | DMA1CH7 interrupt | 0x00000070 |
| 29 | 15 | programmable | ADC1 | ADC1 interrupt | 0x00000074 |
| 30 | 16 | programmable | I2C1_EV | I2C1_EV interrupt | 0x00000078 |
| 31 | 17 | programmable | I2C1_ER | I2C1_ER interrupt | 0x0000007C |
| 32 | 18 | programmable | USART1 | USART1 interrupt | 0x00000080 |
| 33 | 19 | programmable | SPI1 | SPI1 interrupt | 0x00000084 |
| 34 | 20 | programmable | TIM1BRK | TIM1BRK interrupt | 0x00000088 |
| 35 | 21 | programmable | TIM1UP | TIM1UP interrupt | 0x0000008C |
| 36 | 22 | programmable | TIM1TRG | TIM1TRG interrupt | 0x00000090 |
| 37 | 23 | programmable | TIM1CC | TIM1CC interrupt | 0x00000094 |
| 38 | 24 | programmable | TIM2UP | TIM2UP interrupt | 0x00000098 |
| 39 | 25 | programmable | USART2 | USART2 interrupt | 0x0000009C |
| 40 | 26 | programmable | EXTI15_8 | EXTI15_8 interrupt | 0x000000A0 |
| 41 | 27 | programmable | EXTI25_16 | EXTI25_16 interrupt | 0x000000A4 |
| 42 | 28 | programmable | USART3 | USART3 interrupt | 0x000000A8 |
| 43 | 29 | programmable | USART4 | USART4 interrupt | 0x000000AC |
| 44 | 30 | programmable | DMA1_CH8 | DMA1CH8 interrupt | 0x000000B0 |
| 45 | 31 | programmable | USBFS | USBFS interrupt | 0x000000B4 |
| 46 | 32 | programmable | USBFS_WKUP | USBFS Wake-up interrupt | 0x000000B8 |
| 47 | 33 | programmable | PIOC | PIOC interrupt | 0x000000BC |
| 48 | 34 | programmable | OPA | OPA interrupt | 0x000000C0 |
| 49 | 35 | programmable | USBPD | USBPD interrupt | 0x000000C4 |
| 50 | 36 | programmable | USBPD_WKUP | USBPD Wake-up interrupt | 0x000000C8 |
| 51 | 37 | programmable | TIM2CC | TIM2CC global interrupt | 0x000000CC |
| 52 | 38 | programmable | TIM2TRG | TIM2TRG global interrupt | 0x000000D0 |
| 53 | 39 | programmable | TIM2BRK | TIM2BRK global interrupt | 0x000000D4 |
| 54 | 40 | programmable | TIM3 | TIM3 global interrupt | 0x000000D8 |

# 7.4 External Interrupt and Event Controller (EXTI)

## 7.4.1 Overview

Figure 7-1 External interrupt (EXTI) interface block diagram




As can be seen from Figure 7-1, the trigger source of the external interrupt can be either a software interrupt (SWIEVR) or an actual external interrupt channel. The signal of the external interrupt channel will be screened by the edge detect circuit first. Whenever one of the software interrupt or external interrupt signals is generated, it will be output to two AND gate circuits, event enable and interrupt enable, through the OR gate circuit in the figure, as long as an interrupt is enabled or an event is enabled, an interrupt or an event will be generated. six registers of EXTI are accessed by the processor through the HB interface.

## 7.4.2 Wake-up Event

The system can wake up the Sleep mode caused by the WFE command through a wake-up event. The wake-up event is generated by either of the following two configurations.

- Enable an interrupt in a peripheral register, but not enabling this interrupt in the PFIC of the core, and enabling the SEVONPEND bit in the core at the same time. Embodied in EXTI, it is to enable an EXTI interrupt, but not to enable the EXTI interrupt in PFIC, and to enable the SEVONPEND bit at the same time. When the CPU wakes up from WFE, it needs to clear the EXTI interrupt flag bit and the PFIC pending bit.
- Enable an EXTI channel as an event channel eliminates the need for the CPU to clear the interrupt flag bit and the PFIC pending bit after waking up from the WFE.

### 7.4.3 Description

Using an external interrupt requires configuring the corresponding external interrupt channel, i.e. selecting the corresponding trigger edge and enabling the corresponding interrupt. When the set trigger edge appears on the external interrupt channel, an interrupt request will be generated and the corresponding interrupt flag bit will be set. The flag bit can be cleared by writing 1 to the flag bit.

Steps for using external hardware interrupts.

1) Configuration of GPIO operations.
2) Configure the interrupt enable bit (EXTI_INTENR) for the corresponding external interrupt channel.
3) Configuring the trigger edge (EXTI_RTENR or EXTI_FTENR) to select rising edge trigger, falling edge trigger or double edge trigger.
4) Configure EXTI interrupts in the core's PFIC to ensure they can respond correctly.

Steps for using external hardware events.

1) Configuration of GPIO operations.
2) Configure the event enable bit (EXTI_EVENR) for the corresponding external interrupt channel.
3) Configure the trigger edge (EXTI_RTENR or EXTI_FTENR) to select rising edge trigger, falling edge trigger, or double edge trigger.

Using the software interrupt/event steps.

1) Enabling external interrupts (EXTI_INTENR) or external events (EXTI_EVENR).
2) If using interrupt service functions, the EXTI interrupt needs to be set in the core's PFIC.
3) Set the software interrupt trigger (EXTI_SWIEVR), that is, an interrupt will be generated.

### 7.4.4 External Event Mapping

Table 7-2 EXTI Interrupt Mapping

| External interrupt/ Event lines | Mapping event description |
|---|---|
| EXTI0-EXTI23 | Px0-Px23 (x=A/B/C), any of the I/O ports can be enabled for external interrupt/event functions, configured by the AFIO_EXTICRx register. |
| EXTI24 | Enabled on the 2-wire debug interface for PC18 to enable external interrupts or event wake-up |
| EXTI25 | Enabled on the 2-wire debug interface for PC19 to enable external interrupts or event wake-up |
| EXTI26 | PVD event: voltage monitoring threshold exceeded |
| EXTI27 | AWU auto-wakeup event |
| EXTI28 | USB wakeup event |
| EXTI29 | USB PD wakeup event |

## 7.5 Register Description

### 7.5.1 EXTI Registers

Table 7-3 EXTI-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_EXTI_INTENR | 0x40010400 | Interrupt enable register | 0x00000000 |
| R32_EXTI_EVENR | 0x40010404 | Event enable register | 0x00000000 |
| R32_EXTI_RTENR | 0x40010408 | Rising edge trigger enable register | 0x00000000 |

| | | | |
|---|---|---|---|
| R32_EXTI_FTENR | 0x4001040C | Falling edge trigger enable register | 0x00000000 |
| R32_EXTI_SWIEVR | 0x40010410 | Soft interrupt event register | 0x00000000 |
| R32_EXTI_INTFR | 0x40010414 | Interrupt flag register | 0x0000XXXX |

#### 7.5.1.1 Interrupt Enable Register (EXTI_INTENR)

Offset address: 0x00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | MR29 | MR28 | MR27 | MR26 | MR25 | MR24 | MR23 | MR22 | MR21 | MR20 | MR19 | MR18 | MR17 | MR16 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MR15 | MR14 | MR13 | MR12 | MR11 | MR10 | MR9 | MR8 | MR7 | MR6 | MR5 | MR4 | MR3 | MR2 | MR1 | MR0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved | 0 |
| [29:0] | MRx | RW | Enable the interrupt request signal for external interrupt channel x.<br>1: Enable interrupts for this channel.<br>0: Mask interrupts for this channel. | 0 |

#### 7.5.1.2 Event Enable Register (EXTI_EVENR)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | MR29 | MR28 | MR27 | MR26 | MR25 | MR24 | MR23 | MR22 | MR21 | MR20 | MR19 | MR18 | MR17 | MR16 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MR15 | MR14 | MR13 | MR12 | MR11 | MR10 | MR9 | MR8 | MR7 | MR6 | MR5 | MR4 | MR3 | MR2 | MR1 | MR0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved. | 0 |
| [29:0] | MRx | RW | Enable the event request signal for external interrupt channel x.<br>1: Enable events for this channel.<br>0: Mask events for this channel. | 0 |

#### 7.5.1.3 Rising Edge Trigger Enable Register (EXTI_RTENR)

Offset address: 0x08

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | TR29 | TR28 | TR27 | TR26 | TR25 | TR24 | TR23 | TR22 | TR21 | TR20 | TR19 | TR18 | TR17 | TR16 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TR15 | TR14 | TR13 | TR12 | TR11 | TR10 | TR9 | TR8 | TR7 | TR6 | TR5 | TR4 | TR3 | TR2 | TR1 | TR0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved. | 0 |

| | | | | |
|---|---|---|---|---|
| [29:0] | TRx | RW | Enable rising edge triggering for external interrupt channel x.<br>1: Enable rising edge triggering for this channel.<br>0: Disable rising edge triggering for this channel. | 0 |

#### 7.5.1.4 Falling Edge Trigger Enable Register (EXTI_FTENR)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | TR29 | TR28 | TR27 | TR26 | TR25 | TR24 | TR23 | TR22 | TR21 | TR20 | TR19 | TR18 | TR17 | TR16 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TR15 | TR14 | TR13 | TR12 | TR11 | TR10 | TR9 | TR8 | TR7 | TR6 | TR5 | TR4 | TR3 | TR2 | TR1 | TR0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved | 0 |
| [29:0] | TRx | RW | Enable falling edge triggering for external interrupt channel x.<br>1: Enable falling edge triggering for this channel.<br>0: Disable falling edge triggering for this channel. | 0 |

#### 7.5.1.5 Software Interrupt Event Register (EXTI_SWIEVR)

Offset address: 0x10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | SWIER29 | SWIER28 | SWIER27 | SWIER26 | SWIER25 | SWIER24 | SWIER23 | SWIER22 | SWIER21 | SWIER20 | SWIER 19 | SWIER 18 | SWIER 17 | SWIER 16 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SWIER15 | SWIER 14 | SWIER13 | SWIER12 | SWIER11 | SWIER 10 | SWIER 9 | SWIER 8 | SWIER 7 | SWIER 6 | SWIER 5 | SWIER 4 | SWIER 3 | SWIER 2 | SWIER 1 | SWIER 0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved | 0 |
| [29:0] | SWIERx | RW | A software interrupt is set on the corresponding externally triggered interrupt channel. Setting it here causes the interrupt flag bit (EXTI_INTFR) to correspond to the position bit, and if interrupt enable (EXTI_INTENR) or event enable (EXTI_EVENR) is on, then an interrupt or event will be generated. | 0 |

#### 7.5.1.6 Interrupt Flag Register (EXTI_INTFR)

Offset address: 0x14

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | IF29 | IF28 | IF27 | IF26 | IF25 | IF24 | IF23 | IF22 | IF21 | IF20 | IF19 | IF18 | IF17 | IF16 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| IF15 | IF14 | IF13 | IF12 | IF11 | IF10 | IF9 | IF8 | IF7 | IF6 | IF5 | IF4 | IF3 | IF2 | IF1 | IF0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved | 0 |
| [29:0] | IFx | W1 | The interrupt flag bit, this location bit flags that a corresponding external interrupt has occurred. A write of 1 clears this bit. | X |

### 7.5.2 PFIC Registers

Table 7-4 PFIC-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_PFIC_ISR1 | 0xE000E000 | PFIC interrupt enable status register 1 | 0x0000000C |
| R32_PFIC_ISR2 | 0xE000E004 | PFIC interrupt enable status register 2 | 0x00000000 |
| R32_PFIC_ISR3 | 0xE000E008 | PFIC interrupt enable status register 3 | 0x00000000 |
| R32_PFIC_ISR4 | 0xE000E00C | PFIC interrupt enable status register 4 | 0x00000000 |
| R32_PFIC_IPR1 | 0xE000E020 | PFIC interrupt pending status register 1 | 0x00000000 |
| R32_PFIC_IPR2 | 0xE000E024 | PFIC interrupt pending status register 2 | 0x00000000 |
| R32_PFIC_IPR3 | 0xE000E028 | PFIC interrupt pending status register 3 | 0x00000000 |
| R32_PFIC_IPR4 | 0xE000E02C | PFIC interrupt pending status register 4 | 0x00000000 |
| R32_PFIC_ITHRESDR | 0xE000E040 | PFIC interrupt priority threshold configuration register | 0x00000000 |
| R32_PFIC_CFGR | 0xE000E048 | PFIC interrupt configuration register | 0x00000000 |
| R32_PFIC_GISR | 0xE000E04C | PFIC interrupt global status register | 0x00000000 |
| R32_PFIC_VTFIDR | 0xE000E050 | PFIC VTF interrupt ID configuration register | 0x00000000 |
| R32_PFIC_VTFADDRR0 | 0xE000E060 | PFIC VTF interrupt 0 offset address register | 0x00000000 |
| R32_PFIC_VTFADDRR1 | 0xE000E064 | PFIC VTF interrupt 1 offset address register | 0x00000000 |
| R32_PFIC_VTFADDRR2 | 0xE000E068 | PFIC VTF interrupt 2 offset address register | 0x00000000 |
| R32_PFIC_VTFADDRR3 | 0xE000E06C | PFIC VTF interrupt 3 offset address register | 0x00000000 |
| R32_PFIC_IENR1 | 0xE000E100 | PFIC interrupt enable setting register 1 | 0x00000000 |
| R32_PFIC_IENR2 | 0xE000E104 | PFIC interrupt enable setting register 2 | 0x00000000 |
| R32_PFIC_IENR3 | 0xE000E108 | PFIC interrupt enable setting register 3 | 0x00000000 |
| R32_PFIC_IENR4 | 0xE000E10C | PFIC interrupt enable setting register 4 | 0x00000000 |
| R32_PFIC_IRER1 | 0xE000E180 | PFIC interrupt enable clear register 1 | 0x00000000 |
| R32_PFIC_IRER2 | 0xE000E184 | PFIC interrupt enable clear register 2 | 0x00000000 |
| R32_PFIC_IRER3 | 0xE000E188 | PFIC interrupt enable clear register 3 | 0x00000000 |
| R32_PFIC_IRER4 | 0xE000E18C | PFIC interrupt enable clear register 4 | 0x00000000 |
| R32_PFIC_IPSR1 | 0xE000E200 | PFIC interrupt pending setting register 1 | 0x00000000 |
| R32_PFIC_IPSR2 | 0xE000E204 | PFIC interrupt pending setting register 2 | 0x00000000 |
| R32_PFIC_IPSR3 | 0xE000E208 | PFIC interrupt pending setting register 3 | 0x00000000 |
| R32_PFIC_IPSR4 | 0xE000E20C | PFIC interrupt pending setting register 4 | 0x00000000 |
| R32_PFIC_IPRR1 | 0xE000E280 | PFIC interrupt pending clear register 1 | 0x00000000 |
| R32_PFIC_IPRR2 | 0xE000E284 | PFIC interrupt pending clear register 2 | 0x00000000 |
| R32_PFIC_IPRR3 | 0xE000E288 | PFIC interrupt pending clear register 3 | 0x00000000 |
| R32_PFIC_IPRR4 | 0xE000E28C | PFIC interrupt pending clear register 4 | 0x00000000 |
| R32_PFIC_IACTR1 | 0xE000E300 | PFIC interrupt activation status register 1 | 0x00000000 |
| R32_PFIC_IACTR2 | 0xE000E304 | PFIC interrupt activation status register 2 | 0x00000000 |
| R32_PFIC_IACTR3 | 0xE000E308 | PFIC interrupt activation status register 3 | 0x00000000 |
| R32_PFIC_IACTR4 | 0xE000E30C | PFIC interrupt activation status register 4 | 0x00000000 |
| R32_PFIC_IPRIORx | 0xE000E400 | PFIC interrupt priority configuration register | 0x00000000 |
| R32_PFIC_SCTLR | 0xE000ED10 | PFIC system control register | 0x00000000 |

*Note: 1. NMI, EXC, ECALL-M, ECALL-U and BREAKPOINT interrupts are always enabled by default.*

*2. ECALL-M, ECALL-U and BREAKPOINT are all cases of EXC and their status is indicated by bit 3 of the EXC status bit.*

*3. NMI and EXC support interrupt pending clear and set operations, but do not support interrupt enable clear*

*and set operations.*

*4. ECALL-M, ECALL-U and BREAKPOINT do not support interrupt pending clear and set, interrupt enable clear and set operations.*

### 7.5.2.1 PFIC Interrupt Enable Status Register 1 (PFIC_ISR1)

Offset address: 0x00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTENSTA[31:16] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTENST A15 | INTENST A14 | INTENST A13 | INTENST A12 | Reserved | | | | | | | | INTENST A3 | INTENST A2 | Reserved | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | INTENSTA | RO | 12#-31# interrupt current enable status.<br>1: Current numbered interrupt is enabled.<br>0: Current numbered interrupt is not enabled. | 0 |
| [11:4] | Reserved | RO | Reserved. | 0 |
| [3:2] | INTENSTA | RO | 2#-3# interrupt current enable status.<br>1: Current numbered interrupt is enabled.<br>0: Current numbered interrupt is not enabled.<br>*Note: 3#2# interrupts are enabled by default.* | 1 |
| [1:0] | Reserved | RO | Reserved. | 0 |

### 7.5.2.2 PFIC Interrupt Enable Status Register 2 (PFIC_ISR2)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTENSTA[63:48] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTENSTA[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | INTENSTA | RO | 32#-63# interrupt current enable status.<br>1: Current numbered interrupt is enabled.<br>0: Current numbered interrupt is not enabled. | 0 |

### 7.5.2.3 PFIC Interrupt Enable Status Register 3 (PFIC_ISR3)

Offset address: 0x08

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTENSTA[95:80] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTENSTA[79:64] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | INTENSTA | RO | 64#-95# interrupt current enable status. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: Current numbered interrupt is enabled.<br>0: Current numbered interrupt is not enabled. | |

### 7.5.2.4 PFIC Interrupt Enable Status Register 4 (PFIC_ISR4)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | INTENSTA[103:96] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved | 0 |
| [7:0] | INTENSTA | RO | 96#-103# interrupt current enable status.<br>1: Current numbered interrupt is enabled.<br>0: Current numbered interrupt is not enabled. | 0 |

### 7.5.2.5 PFIC Interrupt Pending Status Register 1 (PFIC_IPR1)

Offset address: 0x20

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSTA[31:16] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSTA15 | PENDSTA14 | PENDSTA13 | PENDSTA12 | Reserved | | | | | | | | PENDSTA3 | PENDSTA2 | Reserved | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | PENDSTA | RO | 12#-31# interrupts current pending status:<br>1: Current numbered interrupt is pending;<br>0: Current numbered interrupt is not pending. | 0 |
| [11:4] | Reserved | RO | Reserved. | 0 |
| [3:2] | PENDSTA | RO | 2#-3# interrupts current pending status:<br>1: Current numbered interrupt is pending;<br>0: Current numbered interrupt is not pending. | 0 |
| [1:0] | Reserved | RO | Reserved. | 0 |

### 7.5.2.6 PFIC Interrupt Pending Status Register 2 (PFIC_IPR2)

Offset address: 0x24

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSTA[63:48] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSTA[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | PENDSTA | RO | 32#-63# interrupts current pending status: | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: Current numbered interrupt is pending;<br>0: Current numbered interrupt is not pending. | |

### 7.5.2.7 PFIC Interrupt Pending Status Register 3 (PFIC_IPR3)

Offset address: 0x28

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSTA[95:80] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PENDSTA[79:64] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | PENDSTA | RO | 64#-95# interrupts current pending status:<br>1: Current numbered interrupt is pending;<br>0: Current numbered interrupt is not pending. | 0 |

### 7.5.2.8 PFIC Interrupt Pending Status Register 4 (PFIC_IPR4)

Offset address: 0x2C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | PENDSTA[103:96] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved | 0 |
| [7:0] | PENDSTA | RO | 96#-103# interrupts current pending status:<br>1: Current numbered interrupt is pending;<br>0: Current numbered interrupt is not pending. | 0 |

### 7.5.2.9 PFIC Interrupt Priority Threshold Configuration Register (PFIC_ITHRESDR)

Offset address: 0x40

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | | | | | | | | | | THRESHOLD[15:0] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | THRESHOLD | RW | Interrupt priority threshold setting value.<br>The interrupt priority value lower than the current setting value, when hung, does not perform interrupt service; this register is 0 means the threshold register function is invalid.<br>[7:5]: Priority threshold;<br>[4:0]: Reserved, fixed to 0, write invalid. | 0 |

### 7.5.2.10 PFIC Interrupt Configuration Register (PFIC_CFGR)

Offset address: 0x48

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KEYCODE[15:0] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | SYSRST | Reserved | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | KEYCODE[15:0] | WO | Corresponding to different target control bits, the corresponding security access identification data needs to be written simultaneously in order to be modified, and the readout data is fixed to 0.<br>KEY1 = 0xFA05;<br>KEY2 = 0xBCAF;<br>KEY3 = 0xBEEF. | 0 |
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | SYSRST | WO | System reset (Simultaneous writing to KEY3). Auto clear 0.<br>Writing 1 is valid, writing 0 is invalid.<br>*Note: Same function as the PFIC_SCTLR register SYSRST bit.* | 0 |
| [6:0] | Reserved | RO | Reserved. | 0 |

### 7.5.2.11 PFIC Interrupt Global Status Register (PFIC_GISR)

Offset address: 0x4C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | GPENDSTA | GACTSTA | NESTSTA[7:0] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:10] | Reserved | RO | Reserved. | 0 |
| 9 | GPENDSTA | RO | Are there any interrupts currently on hold.<br>1: Yes; 0: No. | 0 |
| 8 | GACTSTA | RO | Are there any interrupts currently being executed.<br>1: Yes; 0: No. | 0 |
| [7:0] | NESTSTA[7:0] | RO | Current interrupt nesting status, currently supports a maximum of 2 levels of nesting and a maximum hardware stack depth of 2 levels.<br>0x03: Level 2 interrupt in progress.<br>0x01: Level 1 interrupt in progress.<br>0x00: No interrupt occurred.<br>Other: Unlikely cases. | 0 |

### 7.5.2.12 PFIC VTF Interrupt ID Configuration Register (PFIC_VTFIDR)

Offset address: 0x50

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| VTFID3 | | | | | | | | VTFID2 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| VTFID1 | | | | | | | | VTFID0 | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | VTFID3 | RW | Configure the interrupt number of VTF interrupt 3. | 0 |
| [23:16] | VTFID2 | RW | Configure the interrupt number of VTF interrupt 2. | 0 |
| [15:8] | VTFID1 | RW | Configure the interrupt number of VTF interrupt 1. | 0 |
| [7:0] | VTFID0 | RW | Configure the interrupt number of VTF interrupt 0. | 0 |

### 7.5.2.13 PFIC VTF Interrupt 0 Address Register (PFIC_VTFADDRR0)

Offset address: 0x60

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR0[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADDR0[15:1] | | | | | | | | | | | | | | | VTF0EN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | ADDR0 | RW | VTF interrupt 0 service program address bit[31:1], bit0 is 0. | 0 |
| 0 | VTF0EN | RW | VTF interrupt 0 enable bit.<br>1: Enable VTF interrupt 0 channel;<br>0: off. | 0 |

### 7.5.2.14 PFIC VTF Interrupt 1 Address Register (PFIC_VTFADDRR1)

Offset address: 0x64

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR1[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADDR1[15:1] | | | | | | | | | | | | | | | VTF1EN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | ADDR1 | RW | VTF interrupt 1 service program address bit[31:1], bit0 is 0. | 0 |
| 0 | VTF1EN | RW | VTF interrupt 1 enable bit.<br>1: VTF interrupt 1 channel is enabled;<br>0: Off. | 0 |

### 7.5.2.15 PFIC VTF Interrupt 2 Address Register (PFIC_VTFADDRR2)

Offset address: 0x68

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR2[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADDR2[15:1] | | | | | | | | | | | | | | | VTF2EN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | ADDR2 | RW | VTF interrupt 2 service program address bit[31:1], bit0 is 0. | 0 |
| 0 | VTF2EN | RW | VTF interrupt 2 enable bit.<br>1: VTF interrupt 2 channel is enabled;<br>0: Off. | 0 |

### 7.5.2.16 PFIC VTF Interrupt 3 Address Register (PFIC_VTFADDRR3)

Offset address: 0x6C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR3[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADDR3[15:1] | | | | | | | | | | | | | | | VTF3EN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | ADDR3 | RW | VTF interrupt 3 service program address bit[31:1], bit0 is 0. | 0 |
| 0 | VTF3EN | RW | VTF interrupt 3 enable bit.<br>1: VTF interrupt 3 channel is enabled;<br>0: Off. | 0 |

### 7.5.2.17 PFIC Interrupt Enable Setting Register 1 (PFIC_IENR1)

Offset address: 0x100

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTEN[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INTEN15 | INTEN14 | INTEN13 | INTEN12 | Reserved | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | INTEN | WO | 12#-31# interrupt enable control.<br>1: Current number interrupt enable.<br>0: No effect. | 0 |
| [11:0] | Reserved | RO | Reserved. | 0 |

### 7.5.2.18 PFIC Interrupt Enable Setting Register 2 (PFIC_IENR2)

Offset address: 0x104

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTEN[63:48] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INTEN[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | INTEN | WO | 32#-63# interrupt enable control.<br>1: Current number interrupt enable.<br>0: No effect. | 0 |

### 7.5.2.19 PFIC Interrupt Enable Setting Register 3 (PFIC_IENR3)

Offset address: 0x108

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTEN[95:80] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INTEN[79:64] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | INTEN | WO | 64#-95# interrupt enable control.<br>1: Current number interrupt enable.<br>0: No effect. | 0 |

### 7.5.2.20 PFIC Interrupt Enable Setting Register 4 (PFIC_IENR4)

Offset address: 0x10C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | INTEN[103:96] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | INTEN | WO | 96#-103# interrupt enable control.<br>1: Current number interrupt enable.<br>0: No effect. | 0 |

### 7.5.2.21 PFIC Interrupt Enable Clear Register 1 (PFIC_IRER1)

Offset address: 0x180

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTRSET[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INT RST1 | INT RST1 | INT RST1 | INT RST1 | Reserved | | | | | | | | | | | |

| 5 | 4 | 3 | 2 | |
|---|---|---|---|---|

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | INTRSET | WO | 12#-31# interrupt disable control.<br>1: Current number interrupt disable.<br>0: No effect. | 0 |
| [11:0] | Reserved | RO | Reserved. | 0 |

### 7.5.2.22 PFIC Interrupt Enable Clear Register 2 (PFIC_IRER2)

Offset address: 0x184

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTRSET[63:48] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INTRSET[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | INTRSET | WO | 32#-63# interrupt disable control.<br>1: Current number interrupt disable.<br>0: No effect. | 0 |

### 7.5.2.23 PFIC Interrupt Enable Clear Register 3 (PFIC_IRER3)

Offset address: 0x188

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTRSET[95:80] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INTRSET[79:64] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | INTRSET | WO | 64#-95# interrupt disable control.<br>1: Current number interrupt disable.<br>0: No effect. | 0 |

### 7.5.2.24 PFIC Interrupt Enable Clear Register 4 (PFIC_IRER4)

Offset address: 0x18C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | INTRSET[103:96] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | INTRSET | WO | 96#-103# interrupt disable control.<br>1: Current number interrupt disable.<br>0: No effect. | 0 |

### 7.5.2.25 PFIC Interrupt Pending Setting Register 1 (PFIC_IPSR1)

Offset address: 0x200

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSET[31:16] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSET15 | PENDSET14 | PENDSET13 | PENDSET12 | Reserved | | | | | | | | PEND SET3 | PEND SET2 | Reserved | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | PENDSET | WO | 12#-31# interrupt pending settings:<br>1: Current number interrupt pending;<br>0: No effect. | 0 |
| [11:4] | Reserved | RO | Reserved. | 0 |
| [3:2] | PENDSET | WO | 2#-3# interrupt pending settings:<br>1: Current number interrupt pending;<br>0: No effect. | 0 |
| [1:0] | Reserved | RO | Reserved. | 0 |

### 7.5.2.26 PFIC Interrupt Pending Setting Register 2 (PFIC_IPSR2)

Offset address: 0x204

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSET[63:48] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSET[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | PENDSET | WO | 32#-63# interrupt pending settings:<br>1: Current number interrupt pending;<br>0: No effect. | 0 |

### 7.5.2.27 PFIC Interrupt Pending Setting Register 3 (PFIC_IPSR3)

Offset address: 0x208

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSET[95:80] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDSET[79:64] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | PENDSET | WO | 64#-95# interrupt pending settings:<br>1: Current number interrupt pending;<br>0: No effect. | 0 |

### 7.5.2.28 PFIC Interrupt Pending Setting Register 4 (PFIC_IPSR4)

Offset address: 0x20C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | PENDSET[103:96] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | PENDSET | WO | 96#-103# interrupt pending settings:<br>1: Current number interrupt pending;<br>0: No effect. | 0 |

### 7.5.2.29 PFIC Interrupt Pending Clear Register 1 (PFIC_IPRR1)

Offset address: 0x280

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDRST[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | PENDRST14 | Reserved | PENDRST12 | Reserved | | | | | | | | PENDRST3 | PENDRST2 | Reserved | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | PENDRST | WO | 16#-31# interrupt pending cleared:<br>1: Current numbered interrupt clears the pending status;<br>0: No effect. | 0 |
| 15 | Reserved | RO | Reserved. | 0 |
| 14 | PENDRST | WO | 14# interrupt pending cleared:<br>1: Current numbered interrupt clears the pending status;<br>0: No effect. | 0 |
| 13 | Reserved | RO | Reserved. | 0 |
| 12 | PENDRST | WO | 12# interrupt pending cleared:<br>1: Current numbered interrupt clears the pending status;<br>0: No effect. | 0 |
| [11:4] | Reserved | RO | Reserved. | 0 |
| [3:2] | PENDRST | WO | 2#-3# interrupt pending cleared:<br>1: Current numbered interrupt clears the pending status;<br>0: No effect. | 0 |
| [1:0] | Reserved | RO | Reserved. | 0 |

### 7.5.2.30 PFIC Interrupt Pending Clear Register 2 (PFIC_IPRR2)

Offset address: 0x284

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDRST[63:48] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDRST[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | PENDRST | WO | 32#-63# interrupt pending cleared:<br>1: Current numbered interrupt clears the pending status;<br>0: No effect. | 0 |

### 7.5.2.31 PFIC Interrupt Pending Clear Register 3 (PFIC_IPRR3)

Offset address: 0x288

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PENDRST[95:80] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PENDRST[79:64] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | PENDRST | WO | 64#-95# interrupt pending cleared:<br>1: Current numbered interrupt clears the pending status;<br>0: No effect. | 0 |

### 7.5.2.32 PFIC Interrupt Pending Clear Register 4 (PFIC_IPRR4)

Offset address: 0x28C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | PENDRST[103:96] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | PENDSET | WO | 96#-103# interrupt pending cleared:<br>1: Current numbered interrupt clears the pending status;<br>0: No effect. | 0 |

### 7.5.2.33 PFIC Interrupt Activation Status Register 1 (PFIC_IACTR1)

Offset address: 0x300

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IACTS[31:16] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | IACTS14 | Reserved | IACTS12 | Reserved | | | | | | | | IACTS3 | IACTS2 | Reserved | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | IACTS | RO | 16#-31# interrupt execution status.<br>1: Current number interrupt in execution.<br>0: Current number interrupt is not executed. | 0 |
| 15 | Reserved | RO | Reserved. | 0 |
| 14 | IACTS | RO | 14# interrupt execution status.<br>1: Current number interrupt in execution.<br>0: Current number interrupt is not executed. | 0 |
| 13 | Reserved | RO | Reserved. | 0 |
| 12 | IACTS | RO | 12# interrupt execution status.<br>1: Current number interrupt in execution.<br>0: Current number interrupt is not executed. | 0 |
| [11:4] | Reserved | RO | Reserved. | 0 |
| [3:2] | IACTS | RO | 2#-3# interrupt execution status.<br>1: Current number interrupt in execution.<br>0: Current number interrupt is not executed. | 0 |
| [1:0] | Reserved | RO | Reserved. | 0 |

### 7.5.2.34 PFIC Interrupt Activation Status Register 2 (PFIC_IACTR2)

Offset address: 0x304

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IACTS[63:48] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| IACTS[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | IACTS | RO | 32#-63# interrupt execution status.<br>1: Current number interrupt in execution.<br>0: Current number interrupt is not executed. | 0 |

### 7.5.2.35 PFIC Interrupt Activation Status Register 3 (PFIC_IACTR3)

Offset address: 0x308

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IACTS[95:80] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| IACTS[79:64] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | IACTS | RO | 64#-95# interrupt execution status.<br>1: Current number interrupt in execution.<br>0: Current number interrupt is not executed. | 0 |

### 7.5.2.36 PFIC Interrupt Activation Status Register 4 (PFIC_IACTR4)

Offset address: 0x30C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | IACTS[103:96] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | IACTS | WO | 96#-103# interrupt execution status.<br>1: Current number interrupt in execution.<br>0: Current number interrupt is not executed. | 0 |

### 7.5.2.37 PFIC Interrupt Priority Configuration Register (PFIC_IPRIORx) (x=0-63)

Offset address: 0x400-0x4FF

The controller supports 256 interrupts (0-255), each using 8 bits to set the control priority.

| | 31 24 | 23 16 | 15 8 | 7 0 |
|---|---|---|---|---|
| IPRIOR63 | PRIO_255 | PRIO_254 | PRIO_253 | PRIO_252 |
| … | … | … | … | … |
| IPRIORx | PRIO_(4x+3) | PRIO_(4x+2) | PRIO_(4x+1) | PRIO_(4x) |
| … | … | … | … | … |
| IPRIOR0 | PRIO_3 | PRIO_2 | PRIO_1 | PRIO_0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [2047:2040] | IP_255 | RW | Same as IP_0 description. | 0 |
| … | … | … | … | … |
| [31:24] | IP_3 | RW | Same as IP_0 description. | 0 |
| [23:16] | IP_2 | RW | Same as IP_0 description. | 0 |
| [15:8] | IP_1 | RW | Same as IP_0 description. | 0 |
| [7:0] | IP_0 | RW | Number 0 interrupt priority configuration.<br>[7:5]: Priority control bits.<br>If no nesting is configured, no preemption bits;<br>If configured with 2 levels of nesting, bit 7 is the preemption bit;<br>The smaller the priority value, the higher the priority. If interrupts of the same preemption priority hang at the same time, the interrupt with the higher priority is executed first.<br>[4:0]: Reserved, fixed to 0, write invalid. | 0 |

### 7.5.2.38 PFIC System Control Register (PFIC_SCTLR)

Offset address: 0xD10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SYS RST | Reserved | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | RSTE | SET | SEV | WFIT | SLEE | SLEEP | Reser |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | N | EVENT | ONPEND | WFE | PDEEP | ONEXIT | ved |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 31 | SYSRST | WO | System reset, clear 0 automatically. write 1 valid, write 0 invalid, same effect as PFIC_CFGR register. | 0 |
| [30:7] | Reserved | RO | Reserved. | 0 |
| 6 | RSTEN | RW | Core deadlock reset enable switch:<br>1: Off;<br>0: On. | 0 |
| 5 | SETEVENT | WO | Set the event to wake up the WFE case. | 0 |
| 4 | SEVONPEND | RW | When an event occurs or interrupts a pending state, the system can be woken up from after the WFE instruction, or if the WFE instruction is not executed, the system will be woken up immediately after the next execution of the instruction.<br>1: enabled events and all interrupts (Including unenabled interrupts) can wake up the system.<br>0: Only enabled events and enabled interrupts can wake up the system. | 0 |
| 3 | WFITOWFE | RW | Execute the WFI command as if it were a WFE.<br>1: treat the subsequent WFI instruction as a WFE instruction.<br>0: No effect. | 0 |
| 2 | SLEEPDEEP | RW | Low-power mode of the control system.<br>1：deepsleep 0：sleep | 0 |
| 1 | SLEEPONEXIT | RW | System status after control leaves the interrupt service program.<br>1: The system enters low-power mode.<br>0: The system enters the main program. | 0 |
| 0 | Reserved | RO | Reserved. | 0 |

### 7.5.3 Dedicated CSR Registers

A number of Control and Status Registers (CSRs) are defined in the RISC-V architecture to configure or identify or record operational status. In addition to the standard registers defined in the RISC-V Privileged Architecture document, the CH32X035 chip also has a number of vendor-defined registers that need to be accessed using the csr instruction.

*Note: These registers are labeled "MRW, MRO, MRW1" and require the system to be in machine mode to access them.*

#### 7.5.3.1 Interrupt System Control Register (INTSYSCR)

CSR address: 0x804

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | | | GIHWST | Reserved | | | INEST | HWSTKE |

| | | | | |
|---|---|---|---|---|
| | KNEN | | EN | N |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:6] | Reserved | MRO | Reserved. | 0 |
| 5 | GIHWSTKNEN | MRW1 | Global interrupt and hardware stack off enable.<br>*Note: This bit is often used in real time operating systems, when the interrupt switches context, set this bit to turn off the global interrupt and hardware pressure stack out of the stack, when the context switch is completed, after the execution of the interrupt return, the hardware automatically clears this bit.* | 0 |
| [4:2] | Reserved | MRO | Reserved. | 0 |
| 1 | INESTEN | MRW | Interrupt nesting enable:<br>1: Interrupt nesting function enabled;<br>0: Interrupt nesting function disabled. | 0 |
| 0 | HWSTKEN | MRW | Hardware stack enable:<br>1: Hardware stacking function enabled;<br>0: Hardware stacking function disabled. | 0 |

### 7.5.3.2 Exception Entry Base Address Register (MTVEC)

CSR address: 0x305

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BASEADDR[31:16] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BASEADDR[15:2] | | | | | | | | | | | | | | MODE1 | MODE0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:2] | BASEADDR[31:2] | MRW | Interrupt vector table base address. | 0 |
| 1 | MODE1 | MRW | Interrupt vector table identifies patterns.<br>1: Identify by absolute address, support full range, but must jump;<br>0: Identify by jump instruction, limited range, support for non-jump instructions. | 0 |
| 0 | MODE0 | MRW | Interrupt or exception entry address mode selection.<br>1: Address offset based on interrupt number *4;<br>0: Use of a unified entry address. | 0 |

## 7.5.4 Physical Memory Protection Unit (PMP)

To improve system security, a set of physical address access restrictions are defined in RISC-V's architecture, which can set their read, write and execute attributes for the physical memory in the region, with a minimum region length of 4 bytes protected. the PMP unit is always in effect in user mode, and optionally in machine mode, and will generate a system exception interrupt (EXC) if the current memory restrictions are violated. The PMP cell contains four sets of 8-bit configuration registers (32bit) and four sets of address registers which need to be accessed using the csr instruction and are in machine mode.

### 7.5.4.1 PMP Configuration Register (PMPCFG0)

CSR address: 0x3A0

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| pmp3cfg | | | | | | | | pmp2cfg | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| pmp1cfg | | | | | | | | pmp0cfg | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | pmp3cfg | MRW | See pmp0cfg. | 0 |
| [23:16] | pmp2cfg | MRW | See pmp0cfg. | 0 |
| [15:8] | pmp1cfg | MRW | See pmp0cfg. | 0 |
| [7:0] | pmp0cfg | MRW | Bit / Name / Description<br>7 / L / Lock enable, unlockable in machine mode. 1: Latching of relevant registers; 0: No locking.<br>[6:5] / - / Reserved.<br>[4:3] / A / Address alignment and protection area range selection.<br>2 / X / Executable attributes.<br>1 / W / Writable attributes.<br>0 / R / Readable attributes. | 0 |

Among them, address alignment and protection region range selection, for A_ADDR ≤ region < B_ADDR region for memory protection (requiring both A_ADDR and B_ADDR to be 4-byte aligned):

1. If B_ADDR - A_ADDR == $2^2$, the NA4 method is used;
2. If B_ADDR - A_ADDR == $2^{(G+2)}$, G ⩾ 1 and A_ADDR is $2^{(G+2)}$ aligned then the NAPOT method is used;
3. Otherwise the TOR method is used.

| A value | Name | Description |
|---|---|---|
| 00b | OFF | No area to protect |
| 01b | TOR | Top-aligned area protection:<br>Under pmp0cfg, 0≤ region <pmpaddr0;<br>Under pmp1cfg, pmpaddr0≤ region <pmpaddr1;<br>Under pmp2cfg, pmpaddr1≤ region <pmpaddr2;<br>Under pmp3cfg, pmpaddr2≤ region <pmpaddr3.<br>$pmpaddr_{i-1}$ = A_ADDR >> 2;<br>$pmpaddr_i$ = B_ADDR >> 2. |
| 10b | NA4 | Fixed 4-byte area protection.<br>pmp0cfg ~ pmp3cfg corresponds to pmpaddr0 ~ pmpaddr3 as the starting address.<br>$pmpaddr_i$ = A_ADDR >> 2. |
| 11b | NAPOT | Protects the $2^{(G+2)}$ region with G ⩾ 1, at which point A_ADDR is $2^{(G+2)}$ aligned.<br>$pmpaddr_i$ = ((A_ADDR \| ($2^{(G+2)}$-1)) & ~(1<<G+1))>> 2. |

### 7.5.4.2 PMP Address 0 Register (PMPADDR0)

CSR address: 0x3B0

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

| ADDR0[33:18] | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADDR0[17:2] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | ADDR0 | MRW | PMP sets bit[33:2] of address 0, the actual high 2 bits are not used. | 0 |

### 7.5.4.3 PMP Address 1 Register (PMPADDR1)

CSR address: 0x3B1

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR1[33:18] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADDR1[17:2] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | ADDR1 | MRW | PMP sets bit[33:2] of address 1, the actual high 2 bits are not used. | 0 |

### 7.5.4.4 PMP Address 2 Register (PMPADDR2)

CSR address: 0x3B2

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR2[33:18] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADDR2[17:2] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | ADDR2 | MRW | PMP sets bit[33:2] of address 2, the actual high 2 bits are not used. | 0 |

### 7.5.4.5 PMP Address 3 Register (PMPADDR3)

CSR address: 0x3B3

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR3[33:18] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADDR3[17:2] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | ADDR3 | MRW | PMP sets bit[33:2] of address 3, the actual high 2 bits are not used. | 0 |

### 7.5.5 STK Register Description

Table 7-5 STK-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_STK_CTLR | 0xE000F000 | System count control register | 0x00000000 |
| R32_STK_SR | 0xE000F004 | System count status register | 0x00000000 |
| R32_STK_CNTL | 0xE000F008 | System counter low register | 0x00000000 |
| R32_STK_CNTH | 0xE000F00C | System counter high register | 0x00000000 |
| R32_STK_CMPLR | 0xE000F010 | Counting comparison low register | 0x00000000 |
| R32_STK_CMPHR | 0xE000F014 | Counting comparison high register | 0x00000000 |

#### 7.5.5.1 System Count Control Register (STK_CTLR)

Offset address: 0x00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SWIE | Reserved | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | MODE | STRE | STCLK | STIE | STE |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 31 | SWIE | RW | Software interrupt trigger enable (SWI).<br>1: Triggering software interrupts.<br>0: Turn off the trigger.<br>After entering software interrupt, software clear 0 is required, otherwise it is continuously triggered. | 0 |
| [30:5] | Reserved | RO | Reserved. | |
| 4 | MODE | RW | Counting modes:<br>1: Counting down;<br>0: Counting up. | |
| 3 | STRE | RW | Auto-reload count enable bit.<br>1: Re-counting from 0 after counting up to the comparison value.<br>0: Count up to the comparison value and continue counting up, count down to 0 and start counting down again from the maximum value. | |
| 2 | STCLK | RW | Counter clock source selection bit.<br>1: HCLK for time base.<br>0: HCLK/8 for time base. | |
| 1 | STIE | RW | Counter interrupt enable control bit.<br>1: Enable counter interrupt.<br>0: Disable counter interrupt. | |
| 0 | STE | RW | System counter enable control bit.<br>1: Turn on the system counter STK.<br>0: Turn off the system counter STK and the counter stops counting. | 0 |

#### 7.5.5.2 System Count Status Register (STK_SR)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | CNTIF |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | Reserved | RO | Reserved | 0 |
| 0 | CNTIF | RW0 | Count value comparison flag, write 0 to clear, write 1 to invalidate.<br>1: Counting up to the comparison value and down to 0;<br>0: The comparison value is not reached. | 0 |

### 7.5.5.3 System Counter Low Register (STK_CNTL)

Offset address: 0x08

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNT[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CNT[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | CNT[31:0] | RW | The current counter count value is 32 bits lower. | 0 |

*Note: Register STK_CNTL and register STK_CNTH together form the 64-bit system counter.*

### 7.5.5.4 System Counter High Register (STK_CNTH)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNT[63:48] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CNT[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | CNT[63:32] | RW | The current counter count value is 32 bits higher. | 0 |

*Note: Register STK_CNTL and register STK_CNTH together form the 64-bit system counter.*

### 7.5.5.5 Counting Comparison Low Register (STK_CMPLR)

Offset address: 0x10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CMP[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CMP[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | CMP[31:0] | RW | Sets the comparison counter value 32 bits lower. | 0 |

*Note: Register STK_CMPLR and register STK_CMPHR together form the 64-bit counter comparison value.*

### 7.5.5.6 Counting Comparison High Register (STK_CMPHR)

Offset address: 0x14

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CMP[63:48] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CMP[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | CMP[63:32] | RW | Sets the comparison counter value 32 bits higher. | 0 |

*Note: Register STK_CMPLR and register STK_CMPHR together form the 64-bit counter comparison value.*

# Chapter 8 GPIO and Alternate Function (GPIO/AFIO)

The GPIO ports can be configured into various input or output modes, with built-in pull-up resistors that can be turned off, and some GPIOs have built-in pull-down resistors that can be turned off, and can be configured into push-pull function. the GPIO ports can also be alternate into other functions. PA0-PA7, PB0-PB1 and PC0-PC3 support ADC analogue signal input channels 0 to 13. All GPIO pins support controlled pull-up, only PA0-PA15 and PC16-PC17 support controlled pull-down, the remaining pins do not support pull-down. PC14-PC17 support multiple pull-up modes, which are set by the dedicated control registers corresponding to the PD and USB pins respectively.

## 8.1 Main Features

Each pin of the port can be configured to one of the following multiple modes.

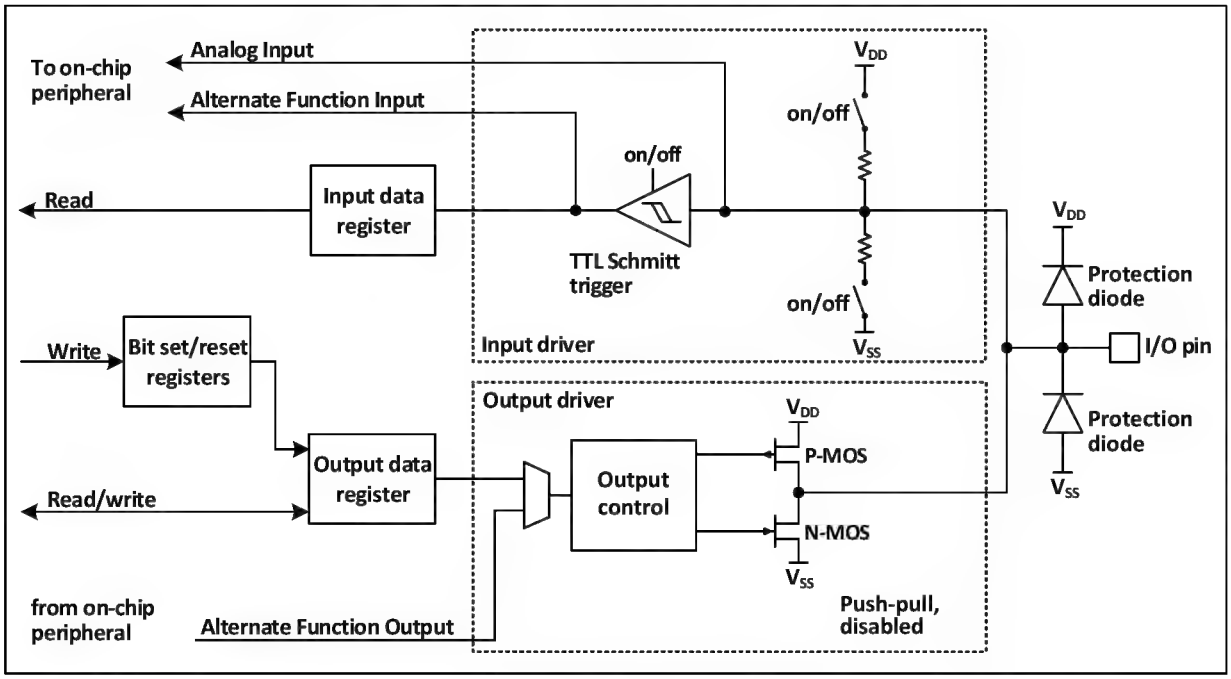
- Floating input
- Pull-up input
- Pull-down input (partial IO)
- Analog input
- Push-pull output
- Inputs and outputs of alternate functions

Many pins have alternate capabilities, and many other peripherals map their output and input channels to these pins. The specific usage of these alternate pins needs to be referred to the individual peripherals, and the content of whether these pins are alternate and remapped is explained in this chapter.

## 8.2 Function Description

### 8.2.1 Overview

Figure 8-1 GPIO module basic structure block diagram




As shown in Figure 8-1 IO port structure, each pin has two protection diodes inside the chip, and the IO port can be internally divided into input and output driver modules. The input driver has a pull-up resistor and a pull-down resistor (Only for some of the IOs) to choose from, and can be connected to peripherals with analogue inputs such

as AD; if the input is to a digital peripheral, it needs to go through a TTL Schmitt trigger and then be connected to a GPIO input register or other multiplexed peripherals. The output driver, on the other hand, has a pair of MOS tubes that can configure the IO port as a push-pull output; the output driver can also be configured internally as to whether the output is controlled by the GPIO or by another peripheral that is multiplexed.

### 8.2.2 GPIO Initialization Function

Just after a reset, the GPIO ports are running in their initial state, when most IO ports are running in the floating input state, but there are also peripheral related pins that are running on the peripheral multiplexing function. Please refer to the relevant section of the pin description for the specific initialization function.

### 8.2.3 External Interrupts

All GPIO ports can be configured with external interrupt input channels, but an external interrupt input channel can only be mapped to at most one GPIO pin, and the serial number of the external interrupt channel must be the same as the bit number of the GPIO port, for example, PA1 (Or PB1, PC1) can only be mapped to EXTI1, and EXTI1 can only accept one of PA1, PB1 or PC1, etc. The mapping of both parties is one-to-one.

### 8.2.4 Alternate Functions

It is important to note that using the alternate function.

- To use the alternate function in the input direction, the port must be configured in alternate input mode, and the pull-down settings can be set according to actual needs.
- To use the alternate function in the output direction, the port must be configured in alternate output mode.
- For bidirectional alternate function, the port must be configured in alternate output mode, when the driver is configured in floating input mode

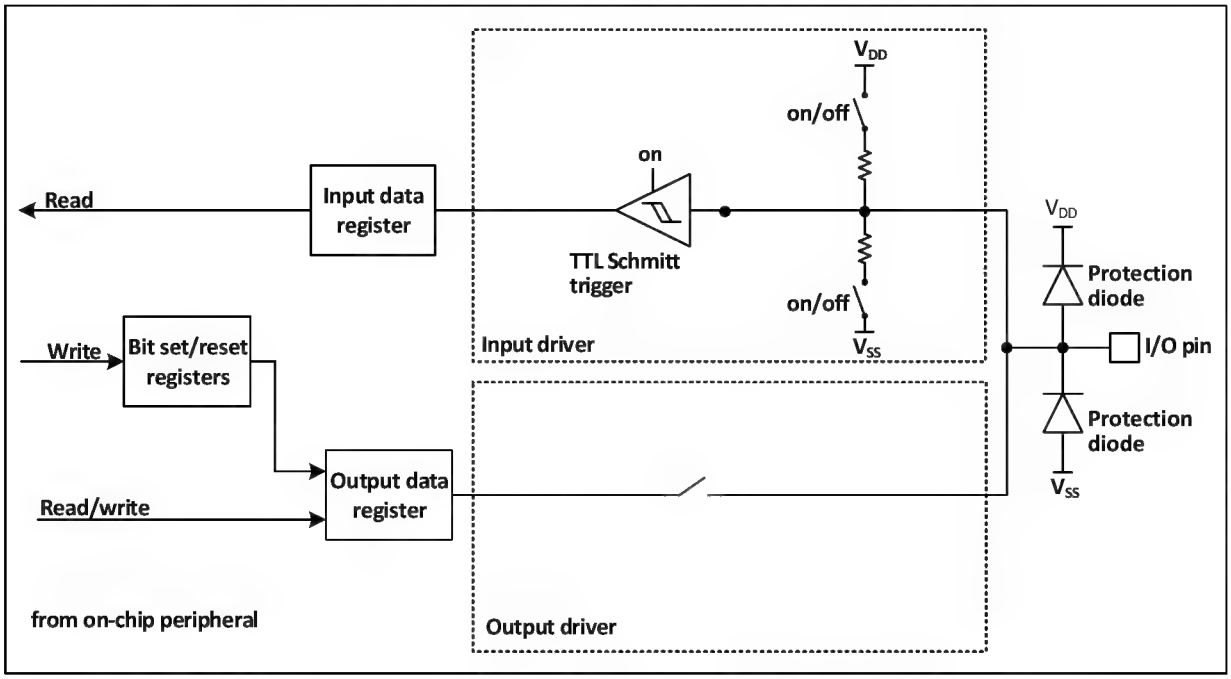
The same I/O port may have multiple peripherals alternate to this pin, so in order to maximize the space for each peripheral, the alternate pins of peripherals can be remapped to other pins in addition to the default alternate pins, avoiding the occupied pins.

### 8.2.5 Locking Mechanism

The locking mechanism locks the configuration of the I/O port. After a specific write sequence, the selected I/O pin configuration will be locked and cannot be changed until the next reset.

### 8.2.6 Input Configuration

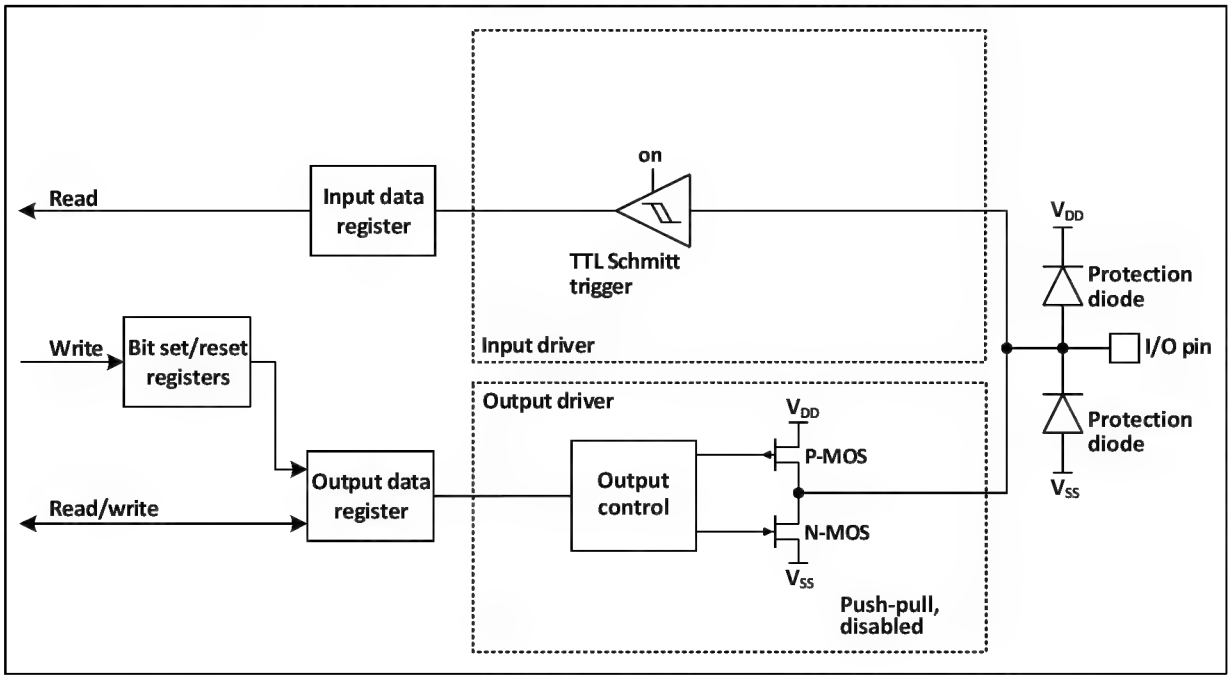
Figure 8-2 GPIO module input configuration structure block diagram




When the IO port is configured in input mode, the output driver is disconnected, the input pull-up and pull-down are selectable, and no alternate functions or analogue inputs are connected. The data on each IO port is sampled at each HB clock into the input data register and the level status of the corresponding pin is obtained by reading the corresponding bit in the input data register.

### 8.2.7 Output Configuration

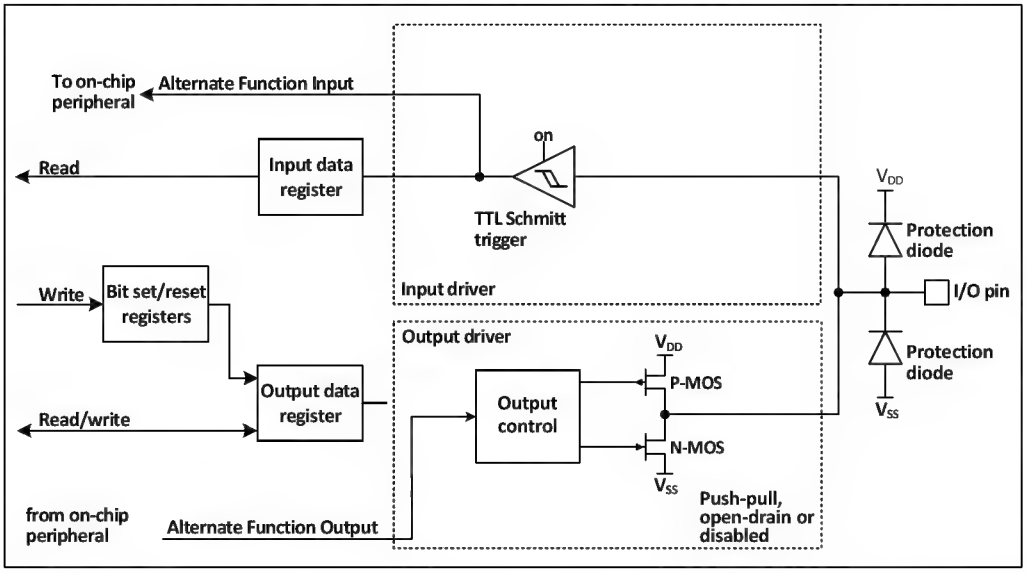
Figure 8-3 GPIO module output configuration structure block diagram



When the IO port is configured in output mode, the output driver is configured in push-pull mode and does not use the multiplexing function. The input driver's pull-up and pull-down resistors are disabled, the TTL Schmitt trigger is activated and the levels appearing on the IO pins will be sampled into the input data registers at each HB clock, so reading the input data registers will give the IO status and access to the output data registers will give the last written value.

### 8.2.8 Alternate Function Configuration

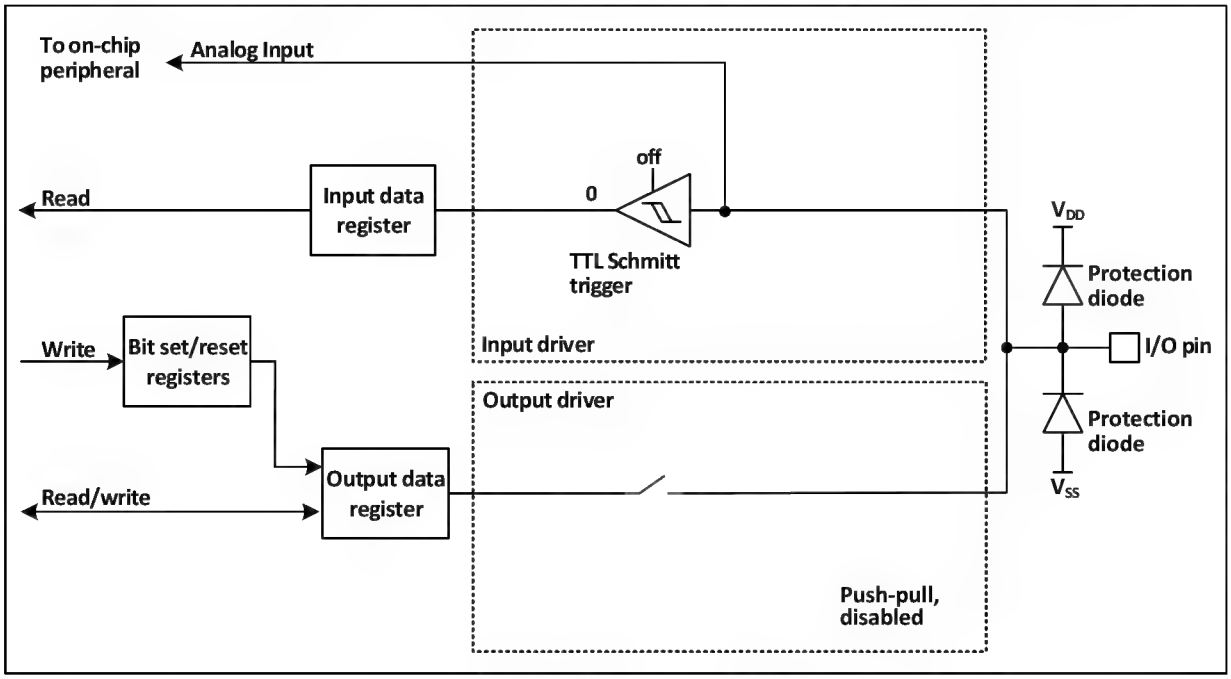
Figure 8-4 The structure of GPIO module when it is alternate by other peripherals




When alternate is enabled, the output drivers are enabled, configured in push-pull mode, or automatically configured in open-drain mode if used for I2C, the Schmitt trigger is turned on, the input and output lines of the alternate function are connected, but the output data registers are disconnected, the levels present on the IO pins will be sampled into the input data registers at each HB clock, and reading the input data registers will give the current status of the IO port; in push-pull mode, reading the output data registers will give the last written value.

### 8.2.9 Analog Input Configuration

Figure 8-5 The configuration structure when the GPIO module is used as an analog input




When the analog input is enabled, the output buffer is disconnected, the input of the Schmitt trigger in the input driver is disabled to prevent the generation of consumption on the I/O port, the pull-up and pull-down resistors are disabled, and the read input data register will always be 0.

### 8.2.10 GPIO Settings for Peripherals

The following table recommends the corresponding GPIO port configuration for each peripheral pin.

Table 8-1 Advanced-control timer (TIM1/2)

| TIM1/2 | Configuration | GPIO configuration |
|---|---|---|
| TIM1/2_CHx | Input capture channel x | Floating input |
| | Output compare channel x | Push-pull alternate output |
| TIM1/2_CHxN | Complementary output channels x | Push-pull alternate output |
| TIM1/2_BKIN | Brake input | Floating input |
| TIM1/2_ETR | Externally triggered clock input | Floating input |

Table 8-2 General-purpose timer (TIM3)

| TIM3 pin | Configuration | GPIO configuration |
|---|---|---|
| TIM3_CHx | Input capture channel x | Floating input |
| | Output comparison channel x | Push-pull alternate output |
| TIM3_ETR | Externally triggered clock input | Floating input |

Table 8-3 Universal synchronous asynchronous serial transceiver (USART)

| USART pin | Configuration | GPIO configuration |
|---|---|---|
| USARTx_TX | Full-duplex mode | Push-pull alternate output |
| | Half-duplex mode | Push-pull alternate output (External plus pull) |

| | | |
|---|---|---|
| USARTx_RX | Full-duplex mode | Floating input or pull-up input |
| | Half-duplex synchronous mode | Not used |
| USARTx_CK | Synchronous mode | Push-pull alternate output |
| USARTx_RTS | Hardware flow control | Push-pull alternate output |
| USARTx_CTS | Hardware flow control | Floating input or pull-up input |

Table 8-4 Serial peripheral interface (SPI) modules

| SPI pin | Configuration | GPIO configuration |
|---|---|---|
| SPIx_SCK | Master mode | Push-pull alternate output |
| | Slave mode | Floating input |
| SPIx_MOSI | Full-duplex Master mode | Push-pull alternate output |
| | Full-duplex Slave mode | Floating input or pull-up input |
| | Simple bi-directional data line/Master mode | Push-pull alternate output |
| | Simple bi-directional data line/Slave mode | Not used |
| SPIx_MISO | Full-duplex Master mode | Floating input or pull-up input |
| | Full-duplex Slave mode | Push-pull alternate output |
| | Simple bi-directional data line/Master mode | Not used |
| | Simple bi-directional data line/Slave mode | Push-pull alternate output |
| SPIx_NSS | Hardware master/slave mode | Floating, pull-up or pull-down input |
| | Hardware master mode/NSS output enable | Push-pull alternate output |
| | Software mode | Not used |

Table 8-5 Internal integrated bus (I2C) module

| I2C pin | Configuration | GPIO configuration |
|---|---|---|
| I2C_SCL | I2C clock | Push-pull alternate output (Automatic open-drain) |
| I2C_SDA | I2C data | Push-pull alternate output (Automatic open drain) |

Table 8-6 USB Host device (USBFS) controller

| USBFS pin | GPIO configuration |
|---|---|
| USBDM/USBDP | After enabling the USB module, the alternate I/O port is automatically connected to the internal USBFS transceiver. |

Table 8-7 Analog-to-digital converters (ADCs)

| ADC pin | GPIO configuration |
|---|---|
| ADC | Analog input |

Table 8-8 OPA function settings

| Pin | Configuration function | GPIO configuration |
|---|---|---|
| OPA_CHP | OPA positive input | Analog input |
| OPA_CHN | OPA negative input | Analog input |
| OPA_OUT | OPA output | Analog input |

Table 8-9 CMP function settings

| Pin | Configuration function | GPIO configuration |
|---|---|---|
| CMP_CHP | CMP positive input | Analog input |

| | | |
|---|---|---|
| CMP_CHN | CMP negative input | Analog input |
| CMP_OUT | CMP output | Push-pull alternate output(1) |

*Note: (1) When the CMP does external output, the GPIO is configured as multiplexed push-pull output, and when the CMP output is an internal channel, it is directly connected to the timer, so there is no need to configure the GPIO at the output.*

Table 8-10 Other I/O function settings

| Pin | Configuration function | GPIO configuration |
|---|---|---|
| MCO | Clock output | Push-pull alternate output |
| EXTI | External interrupt input | Float, pull-up or pull-down input |
| PIOC | Input and output | |

Table 8-11 USB PD / type C controller

| USBPD pin | GPIO configuration |
|---|---|
| CC1/CC2 | When the USBPD module is enabled, the alternate IO port is automatically connected to the internal PD transceiver. |

### 8.2.11 BC Function Configuration

The AFIO_CTLR register includes the control bits for the UDM/UDP pin of the BC interface, the source voltage and the port pull-up mode.

Before the set operation of the BC_VSRC bit of UDM/UDP, it is necessary to operate the PUE[1:0] bits of UDM/UDP to set the pin to the pull-up mode, and this operation can enable the output of the BC protocol source voltage, $V_{BC\_SRC}$; and the read operation of the BC_CMPO bit of UDM/UDP can read the voltage of the pin with the BC protocol reference value $V_{BC\_REF}$ status.

## 8.3 Register Description

### 8.3.1 GPIO Register Description

Unless otherwise specified, the registers of the GPIO must be operated as words (Operate these registers with 32 bits).

Table 8-12 GPIO-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_GPIOA_CFGLR | 0x40010800 | PA port configuration register low (0-7) | 0x44444444 |
| R32_GPIOB_CFGLR | 0x40010C00 | PB port configuration register low (0-7) | 0x44444444 |
| R32_GPIOC_CFGLR | 0x40011000 | PC port configuration register low (0-7) | 0x44444444 |
| R32_GPIOA_CFGHR | 0x40010804 | PA port configuration register high (8-15) | 0x44444444 |
| R32_GPIOB_CFGHR | 0x40010C04 | PB port configuration register high (8-15) | 0x44444444 |
| R32_GPIOC_CFGHR | 0x40011004 | PC port configuration register high (8-15) | 0x44444444 |
| R32_GPIOA_INDR | 0x40010808 | PA port input data register | 0x0000XXXX |
| R32_GPIOB_INDR | 0x40010C08 | PB port input data register | 0x0000XXXX |
| R32_GPIOC_INDR | 0x40011008 | PC port input data register | 0x0000XXXX |
| R32_GPIOA_OUTDR | 0x4001080C | PA port output data register | 0x00000000 |
| R32_GPIOB_OUTDR | 0x40010C0C | PB port output data register | 0x00000000 |
| R32_GPIOC_OUTDR | 0x4001100C | PC port output data register | 0x00000000 |

| | | | |
|---|---|---|---|
| R32_GPIOA_BSHR | 0x40010810 | PA port set/reset register low 16 bits | 0x00000000 |
| R32_GPIOB_BSHR | 0x40010C10 | PB port set/reset register low 16 bits | 0x00000000 |
| R32_GPIOC_BSHR | 0x40011010 | PC port set/reset register low 16 bits | 0x00000000 |
| R32_GPIOA_BCR | 0x40010814 | PA port reset register | 0x00000000 |
| R32_GPIOB_BCR | 0x40010C14 | PB port reset register | 0x00000000 |
| R32_GPIOC_BCR | 0x40011014 | PC port reset register | 0x00000000 |
| R32_GPIOA_LCKR | 0x40010818 | PA port lock configuration register | 0x00000000 |
| R32_GPIOB_LCKR | 0x40010C18 | PB port lock configuration register | 0x00000000 |
| R32_GPIOC_LCKR | 0x40011018 | PC port lock configuration register | 0x00000000 |
| R32_GPIOA_CFGXR | 0x4001081C | PA port configuration register expansion (16-23) bits | 0x44444444 |
| R32_GPIOB_CFGXR | 0x40010C1C | PB port configuration register expansion (16-23) bits | 0x44444444 |
| R32_GPIOC_CFGXR | 0x4001101C | PC port configuration register expansion (16-23) bits | 0x44444444 |
| R32_GPIOA_BSXR | 0x40010820 | PA port set/reset register high 16 bits | 0x00000000 |
| R32_GPIOB_BSXR | 0x40010C20 | PB port set/reset register high 16 bits | 0x00000000 |
| R32_GPIOC_BSXR | 0x40011020 | PC port set/reset register high 16 bits | 0x00000000 |

### 8.3.1.1 Port Configuration Register Low [7:0] (GPIOx_CFGLR) (x=A/B/C)

Offset address: 0x00

| 31 30 | 29 28 | 27 26 | 25 24 | 23 22 | 21 20 | 19 18 | 17 16 |
|---|---|---|---|---|---|---|---|
| CNF7[1:0] | MODE7[1:0] | CNF6[1:0] | MODE6[1:0] | CNF5[1:0] | MODE5[1:0] | CNF4[1:0] | MODE4[1:0] |
| **15 14** | **13 12** | **11 10** | **9 8** | **7 6** | **5 4** | **3 2** | **1 0** |
| CNF3[1:0] | MODE3[1:0] | CNF2[1:0] | MODE2[1:0] | CNF1[1:0] | MODE1[1:0] | CNF0[1:0] | MODE0[1:0] |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30]<br>[27:26]<br>[23:22]<br>[19:18]<br>[15:14]<br>[11:10]<br>[7:6]<br>[3:2] | CNFy[1:0] | RW | (y=0-7), the configuration bits for port x, by which the corresponding port is configured.<br>When in input mode (MODE=00b).<br>00: Analog input mode.<br>01: Floating input mode.<br>10: With pull-up and pull-down mode.<br>11: Reserved.<br>In output mode (MODE>00b).<br>00: General-purpose push-pull output mode.<br>10: Alternate function push-pull output mode. (I2C automatic open drain) | 01b |
| [29:28]<br>[25:24]<br>[21:20]<br>[17:16]<br>[13:12]<br>[9:8]<br>[5:4]<br>[1:0] | MODEy[1:0] | RW | (y=0-7), port x mode selection, configure the corresponding port by these bits.<br>00: Input mode.<br>01/10/11: Output mode. | 00b |

### 8.3.1.2 Port Configuration Register High [15:8] (GPIOx_CFGHR) (x=A/B/C)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNF15[1:0] | | MODE15[1:0] | | CNF14[1:0] | | MODE14[1:0] | | CNF13[1:0] | | MODE13[1:0] | | CNF12[1:0] | | MODE12[1:0] | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CNF11[1:0] | | MODE11[1:0] | | CNF10[1:0] | | MODE10[1:0] | | CNF9[1:0] | | MODE9[1:0] | | CNF8[1:0] | | MODE8[1:0] | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30]<br>[27:26]<br>[23:22]<br>[19:18]<br>[15:14]<br>[11:10]<br>[7:6]<br>[3:2] | CNFy[1:0] | RW | (y=8-15), the configuration bits for port x, by which the corresponding port is configured.<br>When in input mode (MODE=00b).<br>00: Analog input mode.<br>01: Floating input mode.<br>10: With pull-up and pull-down mode.<br>11: Reserved.<br>In output mode (MODE>00b).<br>00: General-purpose push-pull output mode.<br>10: Alternate function push-pull output mode.<br>(I2C automatic open drain) | 01b |
| [29:28]<br>[25:24]<br>[21:20]<br>[17:16]<br>[13:12]<br>[9:8]<br>[5:4]<br>[1:0] | MODEy[1:0] | RW | (y=8-15), port x mode selection, configure the corresponding port by these bits.<br>00: Input mode.<br>01/10/11: Output mode. | 00b |

### 8.3.1.3 Port Input Register (GPIOx_INDR) (x=A/B/C)

Offset address: 0x08

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | IDR23 | IDR22 | IDR21 | IDR20 | IDR19 | IDR18 | IDR17 | IDR16 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| IDR15 | IDR14 | IDR13 | IDR12 | IDR11 | IDR10 | IDR9 | IDR8 | IDR7 | IDR6 | IDR5 | IDR4 | IDR3 | IDR2 | IDR1 | IDR0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved. | 0 |
| [23:0] | IDRy | RO | (y=0-23), the port input data. The value read out is the high and low status of the corresponding bit. | X |

### 8.3.1.4 Port Output Register (GPIOx_OUTDR) (x=A/B/C)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | ODR23 | ODR22 | ODR21 | ODR20 | ODR19 | ODR18 | ODR17 | ODR16 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ODR15 | ODR14 | ODR13 | ODR12 | ODR11 | ODR10 | ODR9 | ODR8 | ODR7 | ODR6 | ODR5 | ODR4 | ODR3 | ODR2 | ODR1 | ODR0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved. | 0 |
| [23:0] | ODRy | RW | (y=0-23), the data output by the port. the IO port outputs the values of these registers externally.<br>For input modes with pull-up and pull-down:<br>0: pull-down input;<br>1: pull-up input. | 0 |

### 8.3.1.5 Port Reset/Set Register (GPIOx_BSHR) (x=A/B/C)

Offset address: 0x10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BR15 | BR14 | BR13 | BR12 | BR11 | BR10 | BR9 | BR8 | BR7 | BR6 | BR5 | BR4 | BR3 | BR2 | BR1 | BR0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BS15 | BS14 | BS13 | BS12 | BS11 | BS10 | BS9 | BS8 | BS7 | BS6 | BS5 | BS4 | BS3 | BS2 | BS1 | BS0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | BRy | WO | (y=0-15), the corresponding OUTDR bit will be cleared for these location bits, writing 0 has no effect. If both the BR and BS bits are set, the BS bit takes effect. | 0 |
| [15:0] | BSy | WO | (y=0-15), for these position bits will make the corresponding OUTDR position bit, writing 0 has no effect. If both the BR and BS bits are set, the BS bit takes effect. | 0 |

### 8.3.1.6 Port Reset Register (GPIOx_BCR) (x=A/B/C)

Offset address: 0x14

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | BR23 | BR22 | BR21 | BR20 | BR19 | BR18 | BR17 | BR16 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BR15 | BR14 | BR13 | BR12 | BR11 | BR10 | BR9 | BR8 | BR7 | BR6 | BR5 | BR4 | BR3 | BR2 | BR1 | BR0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved. | 0 |
| [23:0] | BRy | WO | (y=0-23), the corresponding OUTDR bit will be cleared for these location bits, writing 0 has no effect. | 0 |

### 8.3.1.7 Port Lock Configuration Register (GPIOx_LCKR) (x=A/B/C)

Offset address: 0x18

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

| Reserved | | | | | | | | | | LCKK | LCK23 | LCK22 | LCK21 | LCK20 | LCK19 | LCK18 | LCK17 | LCK16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | |
| LCK15 | LCK14 | LCK13 | LCK12 | LCK11 | LCK10 | LCK9 | LCK8 | LCK7 | LCK6 | LCK5 | LCK4 | LCK3 | LCK2 | LCK1 | LCK0 | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:25] | Reserved | RO | Reserved | 0 |
| 24 | LCKK | RO | The lock key, which can be written in a specific sequence to achieve locking, but which can be read out at any time. It reads 0 to indicate that no locking is in effect, and reads 1 to indicate that locking is in effect.<br>The write sequence for the lock key is: write 1 - write 0 - write 1 - read 0 - read 1. The last step is not necessary, but can be used to confirm that the lock key is active.<br>Any error while writing the sequence will not enable the activation of the lock and the value of LCK[23:0] cannot be changed while the sequence is being written. After the lock is in effect, the port configuration can only be changed after the next reset. | 0 |
| [23:0] | LCKy | RW | (y=0-23), these bits are 1 to indicate locking the configuration of the corresponding port. These bits can only be changed before the LCKK is unlocked. The locked configuration refers to the configuration registers GPIOx_CFGLR, GPIOx_CFGHR and GPIOx_CFGHHR. | 0 |

*Note: After the LOCK sequence is executed for the corresponding port bit, the configuration of the port bit will not be changed again until the next system reset.*

### 8.3.1.8 GPIO Configuration Register Expansion (16-23) Bits (GPIOx_CFGXR) (x=A/B/C)

Offset address: 0x1C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNF23[1:0] | | MODE23[1:0] | | CNF22[1:0] | | MODE22[1:0] | | CNF21[1:0] | | MODE21[1:0] | | CNF20[1:0] | | MODE20[1:0] | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CNF19[1:0] | | MODE19[1:0] | | CNF18[1:0] | | MODE18[1:0] | | CNF17[1:0] | | MODE17[1:0] | | CNF16[1:0] | | MODE16[1:0] | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30]<br>[27:26]<br>[23:22]<br>[19:18]<br>[15:14]<br>[11:10]<br>[7:6]<br>[3:2] | CNFy[1:0] | RW | (y=16-23), the configuration bits for port x, by which the corresponding port is configured.<br>When in input mode (MODE=00b):<br>00: Analogue input mode;<br>01: Floating input mode;<br>10: With pull-up and pull-down mode.<br>11: Reserved.<br>In output mode (MODE>00b):<br>00: General-purpose push-pull output mode;<br>10: Alternate function push-pull output mode; | 01b |

| | | | | |
|---|---|---|---|---|
| | | | (I2C automatic open drain) | |
| [29:28]<br>[25:24]<br>[21:20]<br>[17:16]<br>[13:12]<br>[9:8]<br>[5:4]<br>[1:0] | MODEy[1:0] | RW | (y=16-23), the mode bits for port x, by which the corresponding port is configured.<br>00: Input mode;<br>01/10/11: Output mode. | 00b |

#### 8.3.1.9 Port Set/Reset Register High (16-23) Bits (GPIOx_BSXR) (x=A/B/C)

Offset address: 0x20

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | BR23 | BR22 | BR21 | BR20 | BR19 | BR18 | BR17 | BR16 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | BS23 | BS22 | BS21 | BS20 | BS19 | BS18 | BS17 | BS16 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved. | 0 |
| [23:16] | BRy | WO | (y=16-23), the corresponding OUTDR bit will be cleared for these location bits, writing 0 has no effect. If both the BR and BS bits are set, the BS bit takes effect. | 0 |
| [15:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | BSy | WO | (y=16-23), for these position bits will make the corresponding OUTDR position bit, writing 0 has no effect. If both the BR and BS bits are set, the BS bit takes effect. | 0 |

### 8.3.2 AFIO Register Description

Unless otherwise specified, AFIO registers must be operated as words (Operate these registers with 32 bits).

Table 8-13 List of AFIO-related registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_AFIO_PCFR1 | 0x40010004 | Remap Register 1 | 0x00000000 |
| R32_AFIO_EXTICR1 | 0x40010008 | External interrupt configuration register 1 | 0x00000000 |
| R32_AFIO_EXTICR2 | 0x4001000C | External interrupt configuration register 2 | 0x00000000 |
| R32_AFIO_CTLR | 0x40010018 | Control register | 0x00000045 |

#### 8.3.2.1 Remap Register 1 (AFIO_PCFR1)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | SW_CFG[2:0] | | | PIOC_RM | TIM3_RM [1:0] | | TIM2_RM [2:0] | | | TIM1_RM [2:1] | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TIM1_RM[ | USART4_RM [2:0] | | | USART3_RM | USART2_RM [2:0] | | | USART1_RM [1:0] | | | I2C1_RM [2:0] | | | SPI1_RM [2:0] | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 0] | | [1:0] | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:27] | Reserved | RO | Reserved. | 0 |
| [26:24] | SW_CFG[2:0] | RW | These bits are used to configure the I/O ports for SW function and trace function. SWD (SDI) is the debug interface to access the core. It is always used as a SWD port after system reset.<br>0xx: SWD (SDI) enabled.<br>100: Turn off SWD (SDI), which functions as a GPIO.<br>Others: Invalid. | 0 |
| 23 | PIOC_RM | RW | The PIOC remapping bit, which controls the remapping of PIOC interface 0 to 1 on the GPIO port<br>0: Default ([0]/PC18, [1]/PC19)<br>1: Mapping ([0]/PC7, [1]/PC19) | 0 |
| [22:21] | TIM3_RM[1:0] | RW | The remapping bits of timer 3, which are user readable and writable. It controls the remapping of Timer 3's channels 1 to 2 on the GPIO ports:<br>00: Default mapping (CH1/PA6, CH2/PA7)<br>01: Mapping (CH1/PB4, CH2/PB5)<br>10: Mapping (CH1/PC19, CH2/PC18)<br>11: Mapping (CH1/PA3, CH2/PA4)<br>*Note: Remapping does not affect TIM3_ETR on PD2.* | 0 |
| [20:18] | TIM2_RM[2:0] | RW | Remap bits for timer 2. These bits can be read and written by the user. It controls the mapping of channels 1 to 4 of Timer 2 on the GPIO ports:<br>000: Mapping (CH1/PA0, CH2/PA1, CH3/PA2, CH4/PA3, ETR/PA19, BKIN/PA20, C1N/PA21, C2N/PA22, C3N/PA23)<br>001: Mapping (CH1/PB21, CH2/PB15, CH3/PA2, CH4/PA3, ETR/PA18, BKIN/PA9, C1N/PA12, C2N/PA13, C3N/PA14)<br>010: Mapping (CH1/PA0, CH2/PA1, CH3/PB3, CH4/PB4, ETR/PA19, BKIN/PA20, C1N/PC3, C2N/PA22, C3N/PA23)<br>011: Mapping (CH1/PB21, CH2/PB15, CH3/PB3, CH4/PB4, ETR/PA18, BKIN/PA9, C1N/PA12, C2N/PA13, C3N/PA14)<br>100: Mapping (CH1/PB16, CH2/PB17, CH3/PB18, CH4/PB19, ETR/PC4, BKIN/PC0, C1N/PC1, C2N/PC2, C3N/PC3)<br>101: Mapping (CH1/PC19, CH2/PA12, CH3/PA13, CH4/PC0, ETR/PA2, BKIN/PB4, C1N/PC18, C2N/PB12, C3N/PB3)<br>11x: Mapping (CH1/PC19, CH2/PC14, CH3/PC15, CH4/PC0, ETR/PA2, BKIN/PB4, C1N/PB11, C2N/PB12, C3N/PB3) | 0 |
| [17:15] | TIM1_RM[2:0] | RW | Remap bits for timer 1. These bits can be read and written by the user. It controls the mapping of channels 1 to 4 of Timer 1 on the GPIO ports:<br>000: Mapping (CH1/PB9, CH2/PB10, CH3/PB11, CH4/PC16, ETR/PC17, BKIN/PB5, C1N/PB6, C2N/PB7, C3N/PB8)<br>001: Mapping (CH1/PB9, CH2/PB10, CH3/PB11, CH4/PC16, ETR/PC17, BKIN/PA6, C1N/PA7, | 0 |

| | | | | |
|---|---|---|---|---|
| | | | C2N/PB0, C3N/PB1)<br>010: Mapping (CH1/PB9, CH2/PB10, CH3/PB11, CH4/PB12, ETR/PC18, BKIN/PB5, C1N/PB6, C2N/PB7, C3N/PB8)<br>011: Mapping (CH1/PC0, CH2/PC1, CH3/PC2, CH4/PC3, ETR/PC18, BKIN/PC4, C1N/PC5, C2N/PC6, C3N/PC7)<br>1xx: Mapping (CH1/CH2/CH4/BKIN/C1N/C2N/C3N cannot use, CH3/PC14, ETR/PC15) | |
| [14:12] | USART4_RM[2:0] | RW | Remap bit for USART4. This bit can be read or written by the user. It controls the mapping of the USART4's RX, CTS, TX, CK, and RTS alternate functions on the GPIO ports.<br>000: Mapping (RX/PB1, CTS/PB15, TX/PB0, CK/PB2, RTS/PA8)<br>001: Mapping (RX/PA9, CTS/PA7, TX/PA5, CK/PA6, RTS/PB21)<br>010: mapping (rx/pc17, cts/pb15, tx/pc16, ck/pb2, rts/pa8)<br>011: Mapping (RX/PA10, CTS/PA14, TX/PB9, CK/PB8, RTS/PA13)<br>1x0: mapping (RX/PC19, CTS/PA5, TX/PB13, CK/PA8, RTS/PA6)<br>1x1: Mapping (RX/PC16, CTS/PB15, TX/PC17, CK/PB2, RTS/PA8) | 0 |
| [11:10] | USART3_RM[1:0] | RW | Remap bit for USART3. This bit can be read or written by the user. It controls the mapping of USART2's RX, CTS, TX, CK, RTS multiplexing functions on the GPIO port.<br>00: Mapping (RX/PB4，CTS/PB6，TX/PB3，CK/PB5，RTS/PB7)<br>01: Mapping (RX/PC19，CTS/PB6，TX/PC18，CK/PB5，RTS/PB7)<br>10: Mapping (RX/PB14，CTS/PA3，TX/PA18，CK/PB8，RTS/PA4)<br>11: Mapping (RX/PB17, CTS/PB18, TX/PB16, CK unavailable, RTS/PB19) | 0 |
| [9:7] | USART2_RM[2:0] | RW | Remap bit for USART2. This bit can be read or written by the user. It controls the mapping of USART2's RX, CTS, TX, CK, RTS multiplexing functions on the GPIO port.<br>000: Mapping (RX/PA3, CTS/PA0, TX/PA2, CK/PA4, RTS/PA1)<br>001: Mapping (RX/PA19, CTS/PA1, TX/PA20, CK/PA23, RTS/PA2)<br>010: Mapping (RX/PA16, CTS/PA17, TX/PA15, CK/PA22, RTS/PA21)<br>011: Mapping (RX/PC1, CTS/PC2, TX/PC0, CK/PB20, RTS/PC3)<br>1xx: Mapping (RX/PA16, CTS/PA17, TX/PA15, CK/PA22, RTS/PC3) | 0 |
| [6:5] | USART1_RM[1:0] | RW | Remap bit for USART1. This bit can be read or written by the user. It controls the mapping of USART1's RX, CTS, TX, CK, RTS multiplexing functions on the GPIO port.<br>00: Mapping (RX/PB11, CTS/PC16, TX/PB10, | 0 |

| | | | | |
|---|---|---|---|---|
| | | | CK/PB9, RTS/PC17)<br>01: Mapping (RX/PA11, CTS/PC16, TX/PA10, CK/PB9, RTS/PC17)<br>10: Mapping (RX/PB11, CTS/PA9, TX/PB10, CK/PB5, RTS/PA8)<br>11: Mapping (RX/PB2, CTS/PA13, TX/PA7, CK/PB12, RTS/PA14) | |
| [4:2] | I2C1_RM[2:0] | RW | Remapping of I2C1. This bit can be read or written by the user. It controls the mapping of the SCL and SDA multiplexing functions of I2C1 to the GPIO ports:<br>000: mapping (SCL/PA10, SDA/PA11)<br>001: Mapping (SCL/PA13, SDA/PA14)<br>010: Mapping (SCL/PC16, SDA/PC17)<br>011: Mapping (SCL/PC19, SDA/PC18)<br>1x0: Mapping (SCL/PC17, SDA/PC16)<br>1x1: Mapping (SCL/PC18, SDA/PC19) | 0 |
| [1:0] | SPI1_RM[1:0] | RW | Remapping of SPI1. This bit can be read or written by the user. It controls the mapping of the NSS, CK, MISO and MOSI multiplexing functions of SPI1 to the GPIO ports:<br>00: Mapping (NSS/PA4, CK/PA5, MISO/PA6, MOSI/PA7)<br>01: Mapping (NSS/PB21, CK/PB15, MISO/PA8, MOSI/PA9)<br>10: Mapping (NSS/PA12, CK/PA11, MISO/PA9, MOSI/PA10)<br>11: Mapping (NSS/PC4, CK/PC5, MISO/PC6, MOSI/PC7) | 0 |

### 8.3.2.2 External Interrupt Configuration Register 1 (AFIO_EXTICR1)

Offset address: 0x08

| 31 30 | 29 28 | 27 26 | 25 24 | 23 22 | 21 20 | 19 18 | 17 16 |
|---|---|---|---|---|---|---|---|
| EXTI15 [31:30] | EXTI14 [29:28] | EXTI13 [27:26] | EXTI12 [25:24] | EXTI11 [23:22] | EXTI10 [21:20] | EXTI9 [19:18] | EXTI8 [17:16] |

| 15 14 | 13 12 | 11 10 | 9 8 | 7 6 | 5 4 | 3 2 | 1 0 |
|---|---|---|---|---|---|---|---|
| EXTI7 [15:14] | EXTI6 [13:12] | EXTI5 [11:10] | EXTI4 [9:8] | EXTI3 [7:6] | EXTI2 [5:4] | EXTI1 [3:2] | EXTI0 [1:0] |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30]<br>[29:28]<br>[27:26]<br>[25:24]<br>[23:22]<br>[21:20]<br>[19:18]<br>[17:16]<br>[15:14]<br>[13:12]<br>[11:10]<br>[9:8]<br>[7:6]<br>[5:4] | EXTIx[1:0] | RW | (x=0-15) external interrupt input pin configuration bit. Used to determine to which port pins the external interrupt pins are mapped.<br>00: xth pin of the PA pin.<br>10: xth pin of the PB pin.<br>11: xth pin of the PC pin.<br>Others: Reserved. | 0 |

| | | | | |
|---|---|---|---|---|
| [3:2]<br>[1:0] | | | | |

### 8.3.2.3 External Interrupt Configuration Register 2 (AFIO_EXTICR2)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EXTI23 [15:14] | | EXTI22 [13:12] | | EXTI21 [11:10] | | EXTI20 [9:8] | | EXTI19 [7:6] | | EXTI18 [5:4] | | EXTI17 [3:2] | | EXTI16 [1:0} | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved. | 0 |
| [15:14]<br>[13:12]<br>[11:10]<br>[9:8]<br>[7:6]<br>[5:4]<br>[3:2]<br>[1:0] | EXTIx[1:0] | RW | (x=16-23), external interrupt input pin configuration bit. Used to determine to which port pins the external interrupt pins are mapped.<br>00: xth pin of the PA pin.<br>10: xth pin of the PB pin.<br>11: xth pin of the PC pin.<br>Others: Reserved. | 0 |

### 8.3.2.4 Control Register (AFIO_CTLR)

Offset address: 0x18

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | PB6_FILT_EN | PB5_FILT_EN | PA4_FILT_EN | PA3_FILT_EN | Reserved | | | | UDM_BC_CMPO | UDP_BC_CMPO | UDM_BC_VSRC | UDP_BC_VSRC |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | USBPD_IN_HVT | USB PD_PHY_V33 | USB_IOEN | USB_PHY_V33 | Reserved | | UDP_PUE [1:0] | | UDM_PUE [1:0] | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:28] | Reserved | RO | Reserved. | 0 |
| 27 | PB6_FILT_EN | RW | Control the input filter enable on the PB6 pin:<br>1: Turn on filtering to remove pulses shorter than 10μs;<br>0: Turn off filtering. | 0 |
| 26 | PB5_FILT_EN | RW | Controls the input filter enable on the PB5 pin:<br>1: Turn on filtering to remove pulses shorter than 10μs, which can be used to filter the output square wave of OPA1;<br>0: Turn off filtering. | 0 |
| 25 | PA4_FILT_EN | RW | Controls the input filter enable on the PA4 pin:<br>1: Turn on filtering to remove pulses shorter than 10μs, which can be used to filter the output square wave of OPA2; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: Turn off filtering. | |
| 24 | PA3_FILT_EN | RW | Controls the input filter enable on the PA3 pin:<br>1: Turn on filtering to remove pulses shorter than 10μs, which can be used to filter the output square wave of OPA1;<br>0: Turn off filtering. | 0 |
| [23:20] | Reserved | RW | Reserved. | 0 |
| 19 | UDM_BC_CMPO | RO | PC16/UDM pin BC protocol comparator status:<br>1: PC16 voltage above BC protocol reference $V_{BC\_REF}$;<br>0: PC16 voltage is lower than BC protocol reference $V_{BC\_REF}$. | 0 |
| 18 | UDP_BC_CMPO | RO | PC17/UDP pin BC protocol comparator status:<br>1: PC17 voltage above BC protocol reference $V_{BC\_REF}$;<br>0: PC17 voltage is lower than BC protocol reference $V_{BC\_REF}$. | 0 |
| 17 | UDM_BC_VSRC | RW | PC16/UDM pin BC protocol source voltage enable:<br>1: PC16 outputs BC protocol source voltage $V_{BC\_SRC}$;<br>0: Output is disabled. | 0 |
| 16 | UDP_BC_VSRC | RW | PC17/UDP pin BC protocol source voltage enable:<br>1： PC17 outputs BC protocol source voltage $V_{BC\_SRC}$;<br>0: Output is disabled. | 0 |
| [15:10] | Reserved | RO | Reserved. | 0 |
| 9 | USBPD_IN_HVT | RW | PD pin PC14/PC15 high threshold input mode:<br>1: High threshold input with a typical value of approximately 2.2V, which reduces I/O power consumption during PD communication;<br>0: Normal GPIO threshold input for GPIO applications. | 0 |
| 8 | USBPD_PHY_V33 | RW | PD transceiver PHY pull-up limit configuration bits:<br>1: Direct VDD with output voltage up to VDD, for GPIO applications or PD applications with a VDD voltage of 3.3V;<br>0: Internal simple buck enabled, for PD applications with VDD voltages above 4V.<br>*Note: When PC14/PC15 is used as a PD transceiver, it should be configured as described above; when used as a normal IO, it is recommended that this bit be configured to 1.* | 0 |
| 7 | USB_IOEN | RW | USB multiplexing IO pins enable:<br>1: Enable USB multiplexing;<br>0: Disable as USB.<br>Enabling USB requires, in addition to USB_IOEN set to 1, the setting of: MODE=0 in GPIO configuration register GPIOC_CFGXR corresponding to PC16 and PC17 to select the input mode; for USB host, CNF=10 to select the input mode with up and down pull, PC16 and PC17 in GPIOC_OUTDR corresponding to bit 0 i.e. down pull; for USB device, CNF=10 corresponding to PC17 to select the input mode with up and down For USB devices, PC17 corresponding to CNF=10 selects the input mode with up and down pull, PC17 corresponding to bit 1 in GPIOC_OUTDR selects the up pull, and PC16 corresponding to CNF=01 selects the floating input. | 0 |
| 6 | USB_PHY_V33 | RW | USB transceiver PHY output and pull-up limit configuration bits:<br>1: Direct VDD with output voltage up to VDD, for GPIO applications or USB applications with a VDD voltage of 3.3V; | 1 |

| | | | | |
|---|---|---|---|---|
| | | | 0: LDO buck enabled, limited to approx. 3.3V, for USB applications with VDD voltages above 4V. | |
| [5:4] | Reserved | RO | Reserved. | 0 |
| [3:2] | UDP_PUE[1:0] | RW | Pull-up mode bits for PC17/UDP pins:<br>00: Pull-up disabled;<br>01: Pull-up of approx. 35μs, which can be used for GPIO applications;<br>10: Pull-up 10KΩ, usable for USB at VDD=5V or for forming a voltage divider with a 16KΩ pull-down resistor;<br>11: Pull-up 1.5KΩ, can be used for USB at VDD=3.3V.<br>Only pull-up mode is set here. Pull-up enable or pull-down enable is controlled by the configuration in the GPIO configuration register GPIOx_CFG and the port output register GPIOx_OUTD. | 01b |
| [1:0] | UDM_PUE[1:0] | RW | Pull-up mode bits for PC16/UDM pins:<br>00: pull-up disabled;<br>01: pull-up of approx. 35μs, which can be used for GPIO applications;<br>10: pull-up 10KΩ, can be used for USB at VDD=5V or to form a voltage divider with a 16KΩ pull-down resistor;<br>11: pull-up 1.5KΩ, can be used for USB at VDD=3.3V.<br>Only pull-up mode is set here. Pull-up enable or pull-down enable is controlled by the configuration in the GPIO configuration register GPIOx_CFG and the port output register GPIOx_OUTD. | 01b |

# Chapter 9 Direct Memory Access Control (DMA)

The Direct Memory Access Controller (DMA) provides a high-speed means of data transfer between peripherals and memory or between memory and memory without CPU intervention, data can be moved quickly through the DMA to save CPU resources for other operations. Each channel of the DMA controller is dedicated to managing requests for memory access from one or more peripherals. There is also an arbiter to co-ordinate priorities between the channels.

## 9.1 Main Features

- 8 general-purpose independently configurable channels, with additional dedicated independent channels for USB and USB PD.
- Each channel is directly connected to a dedicated hardware DMA request and supports software triggering
- Support Buffer management with loop
- Request priority between multiple channels can be set by software programming (Very high, high, medium and low) and priority setting is determined by the channel number when equal (The lower the channel number the higher the priority)
- Support peripheral-to-memory, memory-to-peripheral, and memory-to-memory transfers
- Flash memory, SRAM, peripheral SRAM and HB peripherals can be used as access sources and targets
- Programmable number of data transfer bytes: up to 65535

## 9.2 Function Description

### 9.2.1 DMA Channel Processing

**1) Arbitration priority**

DMA requests generated by multiple independent channels are fed to the DMA controller via a logical or structure, and only one channel request is currently responded to. An arbiter inside the module selects the peripheral/memory access to be initiated based on the priority of the channel request.

In software management, the application can configure the priority level for each channel independently by setting the PL[1:0] bits of the DMA_CFGRx register, including 4 levels: highest, high, medium and low. When the software setting levels are the same between channels, the module will be selected according to a fixed hardware priority, with the lower channel number having a higher priority than the higher one.

**2) DMA configuration**

When the DMA controller receives a request signal, it accesses the requested peripheral or memory and establishes a data transfer between the peripheral or memory and the memory. It consists of the following 3 main operation steps.

(1) Fetch data from the memory address indicated by the Peripheral Data Register or the Current Peripheral/Memory Address Register. The start address for the first transfer is the peripheral base address or memory address specified by the DMA_PADDRx or DMA_MADDRx registers.

(2) Store data to the memory address indicated by the Peripheral Data Register or the Current Peripheral/Memory Address Register, and the start address for the first transfer is the peripheral base address or memory address specified by the DMA_PADDRx or DMA_MADDRx registers.

(3) Performs a decrement operation of the value in the DMA_CNTRx register, which indicates the number of

operations currently outstanding for transfer.

Each channel includes 3 types of DMA data transfer methods.

- Peripheral to memory (MEM2MEM=0, DIR=0)
- Memory to peripheral (MEM2MEM=0, DIR=1)
- Memory to memory (MEM2MEM=1)

*Note: The memory-to-memory mode does not require a peripheral request signal. After configuring this mode (MEM2MEM=1), the channel is turned on (EN=1) to start data transfer. This mode does not support cyclic mode.*

The configuration process is as follows.

1) Set the first address of the peripheral register or the memory data address in the memory-to-memory mode (MEM2MEM=1) in the DMA_PADDRx register. This address will be the source or destination address for data transfer when a DMA request occurs.
2) Set the memory data address in the DMA_MADDRx register. When a DMA request occurs, the transferred data will be read from or written to this address.
3) Set the amount of data to be transferred in the DMA_CNTRx register. This value is decremented after each data transfer.
4) Set the priority of the channel in the PL[1:0] bits of the DMA_CFGRx register.
5) Set the direction of data transfer, cyclic mode, incremental mode for peripheral and memory, data width for peripheral and memory, transfer halfway, transfer complete, and transfer error interrupt enable bits in the DMA_CFGRx register.
6) Set the ENABLE bit of the DMA_CCRx register to start channel x.

*Note: The DMA_PADDRx/DMA_MADDRx/DMA_CNTRx registers and the direction of data transfer (DIR), cyclic mode (location), and incremental mode of peripherals and memory (MINC/PINC) control bits in the DMA_CFGRx register can be configured to write only when the DMA channel is turned off.*

**3) Circular mode**

Setting CIRC position 1 of the DMA_CFGRx register enables the cyclic mode function for channel data transfers. In cyclic mode, when the number of data transfers becomes 0, the contents of the DMA_CNTRx register are automatically reloaded to its initial value, and the internal peripheral and memory address registers are reloaded to the initial address values set by the DMA_PADDRx and DMA_MADDRx registers, and DMA operation will continue until the channel is turned off or the DMA mode is turned off.

**4) DMA processing status**

- Transfer half: It corresponds to the hardware setting of HTIFx bit in DMA_INTFR register. The DMA transfer bytes half flag will be generated when the number of DMA transfers is reduced to less than half of the initial set value, and an interrupt will be generated if HTIE is set in the DMA_CCRx register. The hardware uses this flag to alert the application that it can prepare for a new round of data transfers.
- Transfer completion: corresponds to the hardware setting of the TCIFx bit in the DMA_INTFR register. When the number of DMA transfer bytes decreases to 0, the DMA transfer completion flag will be generated, and if TCIE is set in the DMA_CCRx register, an interrupt will be generated.
- Transfer error: corresponds to a hardware set of the TEIFx bit in the DMA_INTFR register. Reading and writing a reserved address area will generate a DMA transfer error. At the same time the module hardware will automatically clear the EN bit of the DMA_CCRx register corresponding to the channel where the error

occurred, and the channel is turned off. If TEIE is set in the DMA_CCRx register, an interrupt will be generated.

When the application queries the DMA channel status, it can first access the GIFx bit of the DMA_INTFR register to determine which channel is currently experiencing a DMA event, and then process the specific DAM event content for that channel.

### 9.2.2 Programmable Total Data Transfer Size/Data Bit Width/Alignment

The total size of the data to be transferred per DMA channel round is programmable up to 65535 times, and the number of pending transfer bytes is indicated in the DMA_CNTRx register. At EN=0, the set value is written, and at EN=1 when the DMA transfer channel is turned on, this register becomes a read-only attribute with a decreasing value after each transfer.

The transferred data fetch values of peripherals and memories support the address pointer auto-increment function with programmable pointer increments. The first transmitted data address they access is stored in the DMA_PADDRx and DMA_MADDRx registers.By setting the PINC bit or MINC position 1 of the DMA_CFGRx register, the peripheral address self-increment mode or memory address self-increment mode can be enabled, respectively. PSIZE[1:0] sets the peripheral address fetch data size and address self-increment size. MSIZE[1:0] sets the memory address to take the data size and address self-increasing small, including three choices: 8-bit, 16-bit, 32-bit. The specific data transfer methods are listed in the following table.

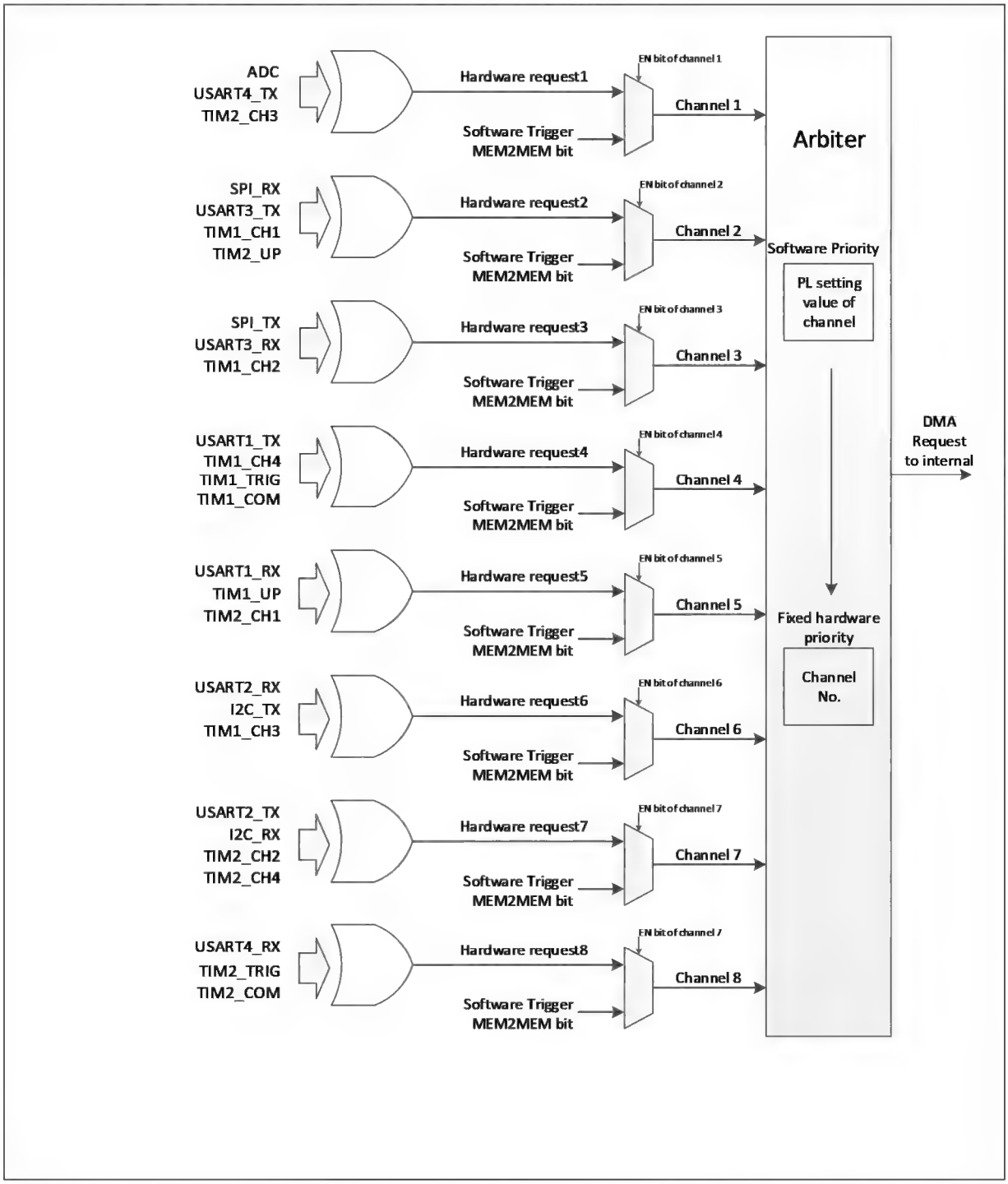
Table 9-1 DMA transfer with different data bit widths (PINC=MINC=1)

| Source bit width | Objectives bit width | Transmission number | Source: address/data | Target: address/data | Transfer operations |
|---|---|---|---|---|---|
| 8 | 8 | 4 | 0x00/B0<br>0x01/B1<br>0x02/B2<br>0x03/B3 | 0x00/B0<br>0x01/B1<br>0x02/B2<br>0x03/B3 | ● The source address increment is aligned with the data bit width set at the source and takes a value equal to the data bit width at the source<br>● The target address increment is aligned with the bit width of the target setup data and takes a value equal to the target data bit width<br>● DMA transfer of data sent to the target based on the principle: the high bit of the data size is not enough to make up 0, the high bit of the data size overflow is removed<br>● Storage data mode: small-end mode, low address stores low bytes, high address stores high bytes |
| 8 | 16 | 4 | 0x00/B0<br>0x01/B1<br>0x02/B2<br>0x03/B3 | 0x00/00B0<br>0x02/00B1<br>0x04/00B2<br>0x06/00B3 | |
| 8 | 32 | 4 | 0x00/B0<br>0x01/B1<br>0x02/B2<br>0x03/B3 | 0x00/000000B0<br>0x04/000000B1<br>0x08/000000B2<br>0x0C/000000B3 | |
| 16 | 8 | 4 | 0x00/B1B0<br>0x02/B3B2<br>0x04/B5B4<br>0x06/B7B6 | 0x00/B0<br>0x01/B2<br>0x02/B4<br>0x03/B6 | |
| 16 | 16 | 4 | 0x00/B1B0<br>0x02/B3B2<br>0x04/B5B4<br>0x06/B7B6 | 0x00/B1B0<br>0x02/B3B2<br>0x04/B5B4<br>0x06/B7B6 | |
| 16 | 32 | 4 | 0x00/B1B0<br>0x02/B3B2<br>0x04/B5B4<br>0x06/B7B6 | 0x00/0000B1B0<br>0x04/0000B3B2<br>0x08/0000B5B4<br>0x0C/0000B7B6 | |
| 32 | 8 | 4 | 0x00/B3B2B1B0<br>0x04/B7B6B5B4<br>0x08/BBBAB9B8<br>0x0C/BFBEBDBC | 0x00/B0<br>0x01/B4<br>0x02/B8<br>0x03/BC | |

| | | | | | |
|---|---|---|---|---|---|
| 32 | 16 | 4 | 0x00/B3B2B1B0<br>0x04/B7B6B5B4<br>0x08/BBBAB9B8<br>0x0C/BFBEBDBC | 0x00/B1B0<br>0x02/B5B4<br>0x04/B9B8<br>0x06/BDBC | |
| 32 | 32 | 4 | 0x00/B3B2B1B0<br>0x04/B7B6B5B4<br>0x08/BBBAB9B8<br>0x0C/BFBEBDBC | 0x00/B3B2B1B0<br>0x04/B7B6B5B4<br>0x08/BBBAB9B8<br>0x0C/BFBEBDBC | |

### 9.2.3 DMA Request Mapping

Figure 9-1 DMA request image




The DMA controller provides 8 channels, each of which corresponds to multiple peripheral requests. By setting the corresponding DMA control bits in the corresponding peripheral registers, the DMA function of each peripheral can be switched on or off independently, with the following correspondence.

| Peripheral | Channel1 | Channel 2 | Channel 3 | Channel 4 | Channel 5 | Channel 6 | Channel 7 | Channel 8 |
|---|---|---|---|---|---|---|---|---|
| ADC | ADC | | | | | | | |
| SPI | | SPI_RX | SPI_TX | | | | | |
| USART1 | | | | USART1_T | USART1_R | | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | X | X | | | |
| USART2 | | | | | | USART2_RX | USART2_TX | |
| USART3 | | USART3_TX | USART3_RX | | | | | |
| USART4 | USART4_TX | | | | | | | USART4_RX |
| I2C | | | | | | I2C_TX | I2C_RX | |
| TIM1 | | TIM1_CH1 | TIM1_CH2 | TIM1_CH4<br>TIM1_TRIG<br>TIM1_COM | TIM1_UP | TIM1_CH3 | | |
| TIM2 | TIM2_CH3 | TIM2_UP | | | TIM2_CH1 | | TIM2_CH2<br>TIM2_CH4 | TIM2_TRIG<br>TIM2_COM |

## 9.3 Register Description

Table 9-2 DMA-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_DMA_INTFR | 0x40020000 | DMA interrupt status register | 0x00000000 |
| R32_DMA_INTFCR | 0x40020004 | DMA interrupt flag clear register | 0x00000000 |
| R32_DMA_CFGR1 | 0x40020008 | DMA channel 1 configuration register | 0x00000000 |
| R32_DMA_CNTR1 | 0x4002000C | DMA channel 1 transfer data number register | 0x00000000 |
| R32_DMA_PADDR1 | 0x40020010 | DMA channel 1 peripheral address register | 0x00000000 |
| R32_DMA_MADDR1 | 0x40020014 | DMA channel 1 memory address register | 0x00000000 |
| R32_DMA_CFGR2 | 0x4002001C | DMA channel 2 configuration register | 0x00000000 |
| R32_DMA_CNTR2 | 0x40020020 | DMA channel 2 transfer data number register | 0x00000000 |
| R32_DMA_PADDR2 | 0x40020024 | DMA channel 2 peripheral address register | 0x00000000 |
| R32_DMA_MADDR2 | 0x40020028 | DMA channel 2 memory address register | 0x00000000 |
| R32_DMA_CFGR3 | 0x40020030 | DMA channel 3 configuration register | 0x00000000 |
| R32_DMA_CNTR3 | 0x40020034 | DMA channel 3 transfer data number register | 0x00000000 |
| R32_DMA_PADDR3 | 0x40020038 | DMA channel 3 peripheral address register | 0x00000000 |
| R32_DMA_MADDR3 | 0x4002003C | DMA channel 3 memory address register | 0x00000000 |
| R32_DMA_CFGR4 | 0x40020044 | DMA channel 4 configuration register | 0x00000000 |
| R32_DMA_CNTR4 | 0x40020048 | DMA channel 4 transfer data number register | 0x00000000 |
| R32_DMA_PADDR4 | 0x4002004C | DMA channel 4 peripheral address register | 0x00000000 |
| R32_DMA_MADDR4 | 0x40020050 | DMA channel 4 memory address register | 0x00000000 |
| R32_DMA_CFGR5 | 0x40020058 | DMA channel 5 configuration register | 0x00000000 |
| R32_DMA_CNTR5 | 0x4002005C | DMA channel 5 transfer data number register | 0x00000000 |
| R32_DMA_PADDR5 | 0x40020060 | DMA channel 5 peripheral address register | 0x00000000 |
| R32_DMA_MADDR5 | 0x40020064 | DMA channel 5 memory address register | 0x00000000 |
| R32_DMA_CFGR6 | 0x4002006C | DMA channel 6 configuration register | 0x00000000 |
| R32_DMA_CNTR6 | 0x40020070 | DMA channel 6 transfer data number register | 0x00000000 |
| R32_DMA_PADDR6 | 0x40020074 | DMA channel 6 peripheral address register | 0x00000000 |
| R32_DMA_MADDR6 | 0x40020078 | DMA channel 6 memory address register | 0x00000000 |
| R32_DMA_CFGR7 | 0x40020080 | DMA channel 7 configuration register | 0x00000000 |
| R32_DMA_CNTR7 | 0x40020084 | DMA channel 7 transfer data number register | 0x00000000 |
| R32_DMA_PADDR7 | 0x40020088 | DMA channel 7 peripheral address register | 0x00000000 |
| R32_DMA_MADDR7 | 0x4002008C | DMA channel 7 memory address register | 0x00000000 |
| R32_DMA_CFGR8 | 0x40020094 | DMA channel 8 configuration register | 0x00000000 |
| R32_DMA_CNTR8 | 0x40020098 | DMA channel 8 transfer data number register | 0x00000000 |
| R32_DMA_PADDR8 | 0x4002009C | DMA channel 8 peripheral address register | 0x00000000 |
| R32_DMA_MADDR8 | 0x400200A0 | DMA channel 8 memory address register | 0x00000000 |

### 9.3.1 DMA Interrupt Status Register (DMA_INTFR)

Offset address: 0x00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TEIF8 | HTIF8 | TCIF8 | GIF8 | TEIF7 | HTIF7 | TCIF7 | GIF7 | TEIF6 | HTIF6 | TCIF6 | GIF6 | TEIF5 | HTIF5 | TCIF5 | GIF5 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TEIF4 | HTIF4 | TCIF4 | GIF4 | TEIF3 | HTIF3 | TCIF3 | GIF3 | TEIF2 | HTIF2 | TCIF2 | GIF2 | TEIF1 | HTIF1 | TCIF1 | GIF1 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 31/27/23/19/15/11/7/3 | TEIFx | RO | Transmission error flag for channel x (x=1/2/3/4/5/6/7/8).<br>1: A transmission error occurred on channel x.<br>0: No transmission error on channel x.<br>Hardware set, software write CTEIFx bit to clear this flag. | 0 |
| 30/26/22/18/14/10/6/2 | HTIFx | RO | Transmission halfway flag for channel x (x=1/2/3/4/5/6/7/8).<br>1: A transmission over half event is generated on channel x.<br>0: No transmission over half on channel x.<br>Hardware set, software write CHTIFx bit to clear this flag. | 0 |
| 29/25/21/17/13/9/5/1 | TCIFx | RO | Transmission completion flag for channel x (x=1/2/3/4/5/6/7/8).<br>1: A transmission completion event is generated on channel x.<br>0: No transmission completion event on channel x.<br>Hardware set, software write CTCIFx bit to clear this flag. | 0 |
| 2824/20/16/12/8/4/0 | GIFx | RO | Global interrupt flag for channel x (x=1/2/3/4/5/6/7/8).<br>1: TEIFx or HTIFx or TCIFx is generated on channel x.<br>0: No TEIFx or HTIFx or TCIFx occurred on channel x.<br>Hardware set, software write CGIFx bit to clear this flag. | 0 |

### 9.3.2 DMA Interrupt Flag Clear Register (DMA_INTFCR)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CTEIF8 | CHTIF8 | CTCIF8 | CGIF8 | CTEIF7 | CHTIF7 | CTCIF7 | CGIF7 | CTEIF6 | CHTIF6 | CTCIF6 | CGIF6 | CTEIF5 | CHTIF5 | CTCIF5 | CGIF5 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CTEIF4 | CHTIF4 | CTCIF4 | CGIF4 | CTEIF3 | CHTIF3 | CTCIF3 | CGIF3 | CTEIF2 | CHTIF2 | CTCIF2 | CGIF2 | CTEIF1 | CHTIF1 | CTCIF1 | CGIF1 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 31/27/23/19/15/11/7/3 | CTEIFx | WO | Clear the transmission error flag for channel x (x=1/2/3/4/5/6/7/8).<br>1: Clear the TEIFx flag in the DMA_INTFR register.<br>0: No effect. | 0 |
| 30/26/22/18/14/10/6/2 | CHTIFx | WO | Clear the transmission halfway flag for channel x (x=1/2/3/4/5/6/7/8).<br>1: Clear the HTIFx flag in the DMA_INTFR register.<br>0: No effect. | 0 |
| 29/25/21/17/13/9/5/1 | CTCIFx | WO | Clear the transmission completion flag for channel x (x=1/2/3/4/5/6/7/8).<br>1: Clear the TCIFx flag in the DMA_INTFR register.<br>0: No effect. | 0 |

| | | | | |
|---|---|---|---|---|
| 2824/20/16/12/8/4/0 | CGIFx | WO | Clear the global interrupt flag for channel x (x=1/2/3/4/5/6/7/8).<br>1: Clear the TEIFx/HTIFx/TCIFx/ GIFx flags in the DMA_INTFR register.<br>0: No effect. | 0 |

### 9.3.3 DMA Channel x Configuration Register (DMA_CFGRx)(x=1/2/3/4/5/6/7/8)

Offset address: 0x08

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | MEM2MEM | PL[1:0] | | MSIZE[1:0] | | PSIZE[1:0] | | MINC | PINC | CIRC | DIR | TEIE | HTIE | TCIE | EN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| 14 | MEM2MEM | RW | Memory-to-memory mode enable.<br>1: Enable memory-to-memory data transfer mode.<br>0: Disable memory-to-memory data transfer mode. | 0 |
| [13:12] | PL[1:0] | RW | Channel priority setting.<br>00: low; 01: medium.<br>10: High; 11: Very high. | 0 |
| [11:10] | MSIZE[1:0] | RW | Memory address data width setting.<br>00: 8-bit; 01: 16-bit.<br>10: 32-bit; 11: Reserved. | 0 |
| [9:8] | PSIZE[1:0] | RW | Peripheral address data width setting.<br>00: 8-bit; 01: 16-bit.<br>10: 32-bit; 11: Reserved. | 0 |
| 7 | MINC | RW | Memory address incremental increment mode enable.<br>1: Enable incremental memory address increment operation.<br>0: Memory address remains unchanged operation. | 0 |
| 6 | PINC | RW | Peripheral address incremental increment mode enable.<br>1: Enable incremental increment operation of the peripheral address.<br>0: Peripheral address remains unchanged operation. | 0 |
| 5 | CIRC | RW | DMA channel cyclic mode enable.<br>1: Enables cyclic operation.<br>0: Perform a single operation. | 0 |
| 4 | DIR | RW | Data transfer direction.<br>1: Read from memory.<br>0: Read from peripheral. | 0 |
| 3 | TEIE | RW | Transmission error interrupt enable control.<br>1: Enable transmission error interrupt.<br>0: Disable transmission error interrupt. | 0 |
| 2 | HTIE | RW | Transmission over half interrupt enable control.<br>1: Enable the transmission over half interrupt.<br>0: Disable the transmission over half interrupt. | 0 |
| 1 | TCIE | RW | Transmission completion interrupt enable control.<br>1: Enable the transmission completion interrupt.<br>0: Disable the transmission completion interrupt. | 0 |
| 0 | EN | RW | Channel enable control.<br>1: Channel on; 0: Channel off.<br>When a DMA transfer error occurs, the hardware | 0 |

| | | | | |
|---|---|---|---|---|
| | | | automatically clears this bit to 0 and shuts down the channel. | |

### 9.3.4 DMA Channel x Number of Data Register (DMA_CNTRx) (x=1/2/3/4/5/6/7/8)

Offset address: 0x0C + (x-1)*20

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| NDT[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved. | 0 |
| [15:0] | NDT[15:0] | RW | Number of data transfers, range 0-65535.<br>Indicates the number of remaining pending transfer bytes (The register content is decremented after each DMA transfer).<br>When the channel is in cyclic mode, the contents of the register will be automatically reloaded to the previously configured value. | 0 |

### 9.3.5 DMA Channel x Peripheral Address Register (DMA_PADDRx) (x=1/2/3/4/5/6/7/8)

Offset address: 0x10 + (x-1)*20

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PA[31:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | PA[31:0] | RW | Peripheral base address, which serves as the source or destination address for peripheral data transfer.<br>When PSIZE[1:0]='01' (16 bits), the module automatically ignores bit0 and the operation address is automatically 2-byte aligned; when PSIZE[1:0]='10' (32 bits), the module automatically ignores bit[1:0] and the operation address is automatically 4-byte aligned. | 0 |

*Note: This register can only be changed when EN=0 and cannot be written when EN=1.*

### 9.3.6 DMA Channel x Memory Address Register (DMA_MADDRx) (x=1/2/3/4/5/6/7/8)

Offset address: 0x14 + (x-1)*20

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MA[31:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | MA[31:0] | RW | The memory data address, which serves as the source or destination address for data transfers.<br>When MSIZE[1:0]='01' (16 bits), the module automatically ignores bit0, and the operation address is automatically 2-byte aligned; when MSIZE[1:0]='10' (32 bits), the module automatically ignores bit[1:0], and the operation address is automatically 4-byte aligned. | 0 |

*Note: This register can only be changed when EN=0 and cannot be written when EN=1.*

# Chapter 10 Analog-to-digital Converter (ADC)

The ADC module contains a 12-bit successive approximation type analogue-to-digital converter. It supports 14 external channels and 1 internal signal source sampling source. Single conversion and continuous conversion of channels, automatic scan mode between channels, intermittent mode, external trigger mode and double sampling can be accomplished. The analog watchdog function allows monitoring whether the channel voltage is within the threshold range.

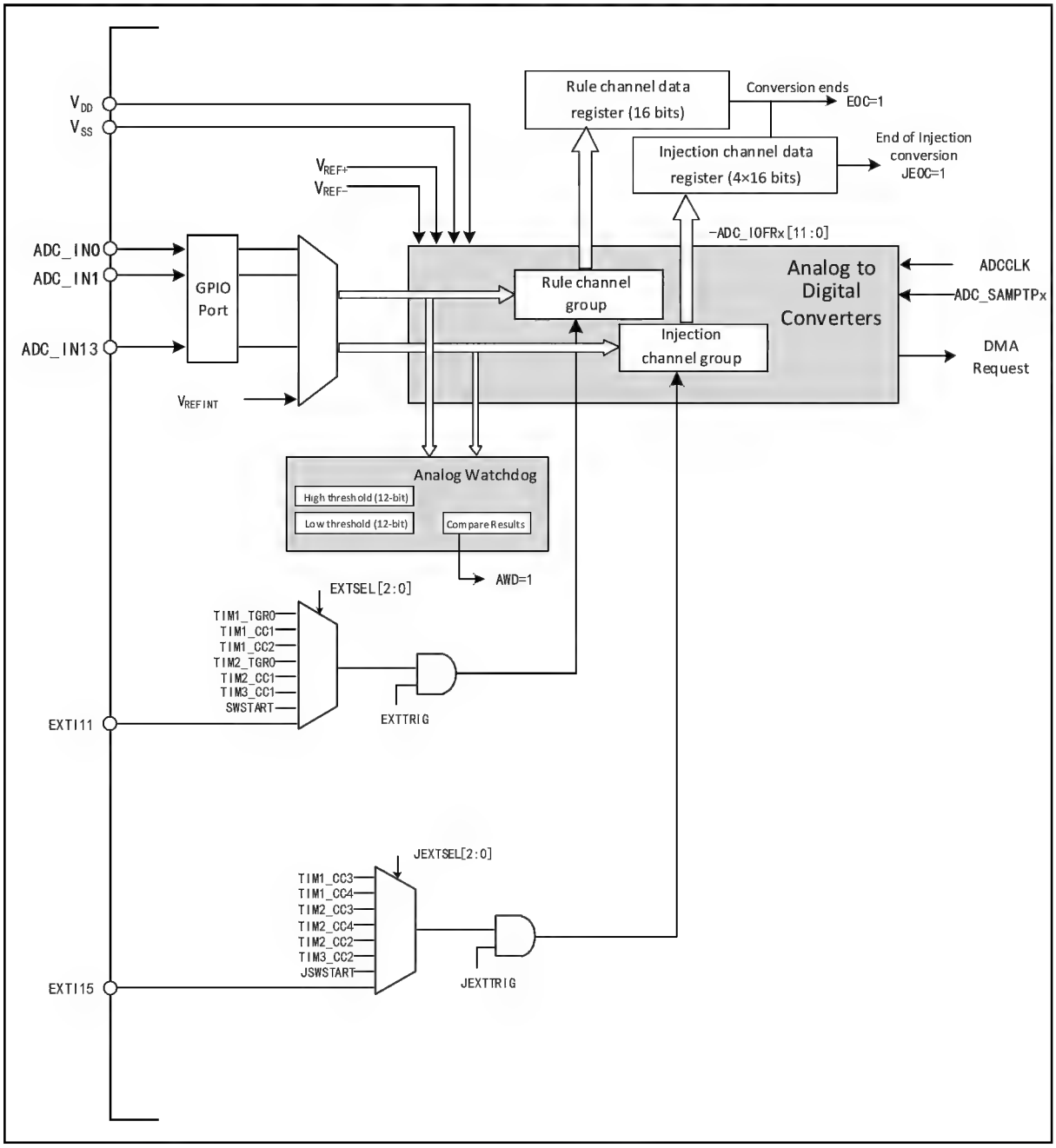
## 10.1 Main Features

- 12-bit resolution
- Supports 14 external channels and 1 internal signal sources for sampling
- Multiple sampling conversion methods for multiple channels: single, continuous, scan, trigger, intermittent, etc.
- Data alignment modes: left-aligned (Rule groups only), right-aligned
- Sampling time can be programmed separately by channel
- Both rule conversion and injection conversion support external triggering
- Analog watchdog to monitor channel voltage ADC channel input range: $0 \leqslant V_{IN} \leqslant V_{DD}$

# 10.2 Functional Description

## 10.2.1 Module Structure

Figure 10-1 ADC module block diagram




## 10.2.2 ADC Configuration

### 1) Module power-up

An ADON bit of 1 in the ADC_CTLR2 register indicates that the ADC module is powered up. When the ADC module enters the power-up state (ADON=1) from the power-down mode (ADON=0), a delay period $t_{STAB}$ is required for the module stabilization time. After that, the ADON bit is written to 1 again and is used as the start signal for software to start the ADC conversion. By clearing the ADON bit to 0, the current conversion can be terminated and the ADC module placed in power-down mode, a state in which the ADC consumes almost no

power.

**2) Sampling clock**

The module's register operations are based on the HCLK (HB bus) clock, whose conversion unit's clock reference ADCCLK is synchronized to HCLK and is configured by dividing the frequency in the CLK_DIV[3:0] field of the ADC_CTLR3 register.

**3) Channel configuration**

The ADC module provides 14 channel sampling sources. They can be configured into two types of conversion groups: regular groups and injection groups. to achieve a group conversion consisting of a series of conversions in any order on any number of channels.

Conversion group:

- Rule group: consists of up to 16 conversions. The rule channels and their conversion order are set in the ADC_RSQRx register. The total number of conversions in the rule group should be written to L[3:0] in the ADC_RSQR1 register.
- Injection group: consists of up to 4 conversions. The injection channels and the order of their conversions are set in the ADC_ISQR register. The total number of conversions in the injection group should be written in JL[1:0] of the ADC_ISQR register.

*Note: If the ADC_RSQRx or ADC_ISQR registers are changed during conversion, the current conversion is terminated and a new start signal is sent to the ADC to convert the newly selected group.*

1 internal channel:

- $V_{REFINT}$ internal reference voltage: connected to ADC_IN15 channel.

**4) Programmable sampling time**

The ADC uses several ADCCLK cycles to sample the input voltage. The number of sampling cycles for a channel can be changed using the SMPx[2:0] bits in the ADC_SAMPTR1 and ADC_SAMPTR2 registers. Each channel can be sampled separately using a different time.

The total conversion time is calculated as follows.

$$T_{CONV} = \text{sampling time} + 13T_{ADCCLK}$$

The ADC's rule channel conversion supports the DMA function. The value of the rule channel conversion is stored in a data-only register, ADC_RDATAR. To prevent the data in ADC_RDATAR register from being fetched in time when multiple rule channels are converted in succession, the DMA function of ADC can be enabled. The hardware will generate a DMA request at the end of the conversion of a rule channel (EOC set) and transfer the converted data from the ADC_RDATAR register to the user-specified destination address.

After the channel configuration of the DMA controller module is completed, write DMA position 1 of the ADC_CTLR2 register to enable the DMA function of the ADC.

*Note: Injection group conversion does not support DMA function.*

**5) Data alignment**

The ALIGN bit in the ADC_CTLR2 register selects the alignment of the ADC converted data storage. 12-bit data supports left-aligned and right-aligned modes.

The data register ADC_RDATAR of the rule group channel holds the actual converted 12-bit digital value; while the data register ADC_IDATARx of the injection group channel is the actual converted data minus the value

written after the offset defined in the ADC_IOFRx register, there will be positive and negative cases, so there are sign bits (SIGNB).

*Note: Data left alignment applies only when rule groups are used.*

Figure 10-2 Data left alignment

Rule group data register

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D4 | D2 | D1 | D0 | 0 | 0 | 0 | 0 |

Figure 10-3 Data right alignment

Rule group data register

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

Inject group data register

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SIGNB | SIGNB | SIGNB | SIGNB | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

### 10.2.3 External Trigger Source

The ADC conversion start event can be triggered by an external event. If the EXTTRIG or JEXTTRIG bits of the ADC_CTLR2 register are set, the conversion of a rule group or injection group channel can be triggered by an external event, respectively. In this case, the configuration of EXTSEL[2:0] and JEXTSEL[2:0] bits determines the external event source for the rule group and injection group.

Table 10-1 External trigger sources for rule group channels

| EXTSEL[2:0] | Trigger source | Type |
|---|---|---|
| 000 | TRGO event of timer 1 | Internal signal from on-chip timer |
| 001 | CC1 event of timer 1 | |
| 010 | CC2 event of timer 1 | |
| 011 | TRGO event of timer 2 | |
| 100 | CC1 event of timer 2 | |
| 101 | CC1 event of timer 3 | |
| 110 | EXTI line 11 | From external pins |
| 111 | SWSTART position 1 software trigger | Software control bits |

Table 10-2 External trigger sources for injection group channels

| JEXTSEL[2:0] | Trigger source | Type |
|---|---|---|
| 000 | CH3 event of timer 1 | Internal signal from on-chip timer |
| 001 | CH4 event of timer 1 | |
| 010 | CH3 event of timer 2 | |
| 011 | CH4 event of timer 2 | |
| 100 | CC2 event of timer 2 | |
| 101 | CC2 event of timer 3 | |
| 110 | EXTI line 15 | From external pins |
| 111 | JSWSTART position 1 software trigger | Software control bits |

### 10.2.4 Conversion Mode

Table 10-3 Conversion mode combinations

| ADC_CTLR1 and ADC_CTLR2 register control bits | | | | | ADC conversion mode |
|---|---|---|---|---|---|
| CONT | SCAN | DISCEN/JDISCEN | IAUTO | Start event | |
| 0 | 0 | 0 | 0 | ADON position 1 | Single single-channel mode: A rule channel performs a single conversion. |
| | | | | External trigger method | Single single-channel mode: A single conversion is performed on one of the rule channels or injection channels. |
| | 1 | 0 | 0 | ADON position 1 or external trigger method | Single scan mode: performs a single conversion of all selected rule group channels (ADC_RSQRx) or all injection group channels (ADC_ISQR) one by one in sequence.<br>Trigger injection method: When the rule group channel conversion process can be inserted into the injection group channel all conversion, and then continue the rule group channel conversion afterwards; but the rule group channel conversion will not be inserted when converting the injection group channel. |
| | | | 1 | ADON position 1 or external trigger method | Single scan mode: performs a single conversion of all selected rule group channels (ADC_RSQRx) or all injection group channels (ADC_ISQR) one by one in sequence.<br>Automatic injection method: After the rule group channel is converted, the injection group channel is automatically converted. |
| | 0 | 1 (DISCEN and JDISCEN cannot be 1 at the same time) | 0 | External trigger method | Single intermittent mode: Each time an event is started, a short sequence (DISCNUM[2:0] defined number) of channel number transitions is executed and cannot be restarted until all selected channel transitions are completed.<br>*Note: The DISCEN and JDISCEN control bits are selected for the rule group and injection group respectively, and the intermittent mode cannot be configured for the rule group and injection group at the same time.* |
| | | | 1 | - | Disable this mode. |
| | 1 | 1 | X | - | No such mode. |
| 1 | 0 | 0 | 0 | ADON position 1 or external trigger method | Continuous single channel/scan mode: repeat a new round of transitions at the end of each round until CONT clears 0 to terminate. |
| | 1 | 0 | 0 | | |
| | | | 1 | | |

*Note: The external trigger events for rule groups and injection groups are different, and the 'ACON' bit can only initiate rule group channel conversion, so the initiation events for rule group and injection group channel conversion are independent.*

**1) Single single-channel conversion mode**

In this mode, only one conversion is executed for the current 1 channel. This mode performs conversion for the channel that is sorted 1st in the rule group or injection group, where it is initiated by setting ADON position 1 of the ADC_CTLR2 register (For rule channels only) or can be initiated by external trigger (For rule channels or injection channels). Once the conversion of the selected channel is completed it will.

If the conversion is for a rule group channel, the conversion data is stored in the 16-bit ADC_RDATAR register, the EOC flag is set, and an ADC interrupt is triggered if the EOCIE bit is set.
If the conversion is for an injection group channel, the conversion data is stored in the 16-bit ADC_IDATAR1 register, the EOC and JEOC flags are set, and an ADC interrupt is triggered if the JEOCIE or EOCIE bit is set.

**2) Single scan mode conversion**

The ADC scan mode is entered by setting the SCAN bit of the ADC_CTLR1 register to 1. This mode is used to scan a group of analog channels and perform a single conversion for all channels selected by ADC_RSQRx register (For regular channels) or ADC_ISQR (For injection channels) one by one, and the next channel in the same group is converted automatically when the current channel conversion is finished.
In the scan mode, there is a subdivision into triggered injection mode and automatic injection mode depending on the status of the IAUTO bit.

- Trigger injection

IAUTO bit is 0. When the trigger event of injection group channel conversion occurs during the scanning of rule group channels, the current conversion is reset and the sequence of injection channels is performed in a single scan, and the last interrupted rule group channel conversion is resumed after all selected injection group channel scanning conversions are completed.
If a rule channel start event occurs while the injection group channel sequence is currently being scanned, the injection group conversion is not interrupted, but the rule sequence conversion is executed again after the injection sequence conversion is completed.
*Note: When using triggered injection conversions, you must ensure that the interval between triggered events is longer than the injection sequence. For example, if the overall time to complete the conversion of the injection sequence takes 28 ADCCLK, then the minimum value of the event interval to trigger the injection channel is 29 ADCCLK.*

- Auto-injection

The IAUTO bit is set to 1, and conversion of the selected channel of the injection group is performed automatically after scanning all the channels selected by the rule group for conversion. This approach can be used to convert up to 20 conversion sequences in the ADC_RSQRx and ADC_ISQR registers.
In this mode, external triggering of the injection channel must be disabled (JEXTTRIG=0).
*Note: For ADC clock prescaler factor (CLK_DIV[3:0]) of 4 to 8, 1 ADCCLK interval is automatically inserted when switching from rule conversion to injection sequence or from injection conversion to rule sequence; when ADC clock prescaler factor is 2, there is a delay of 2 ADCCLK intervals.*

**3) Single intermittent mode conversion**

The intermittent mode of the rule group or injection group is entered by setting the DISCEN or JDISCEN bit of the ADC_CTLR1 register to 1. This mode differs from scanning a complete set of channels in scan mode, but divides a set of channels into multiple short sequences, and each external trigger event will perform a short sequence of scan transitions.
The length of the short sequence n (n<=8) is defined in DISCNUM[2:0] of ADC_CTLR1 register, when DISCEN is 1, it is the interrupted mode of the rule group, and the total length to be converted is defined in RLEN[3:0] of ADC_RSQR1 register; when JDISCEN is 1, it is the interrupted mode of the injection group, and the total length to be converted is defined in ILEN[1:0] of ADC_ISQR register. It is not possible to set both the rule group and the injection group to intermittent mode.
Example of rule group intermittent mode.

DISCEN=1, DISCNUM[2:0]=3, L [3:0]=8, channels to be converted = 1, 3, 2, 5, 8, 4, 10, 6

The 1st external trigger: conversion sequence is: 1, 3, 2

The 2nd external trigger: conversion sequence is: 5, 8, 4

The 3rd external trigger: conversion sequence is: 10, 6, while generating EOC events

The 4th external trigger: conversion sequence is: 1, 3, 2

Examples of intermittent patterns injected into groups.

JDISCEN=1, DISCNUM[2:0]=1, JL[1:0]=3, channel to be converted=1, 3, 2

The 1st external trigger: conversion sequence is: 1

The 2nd external trigger: the conversion sequence is: 3

The 3rd external trigger: conversion sequence is: 2, generating both EOC and JEOC events

The 4th external trigger: conversion sequence is: 1

*Note: 1. When converting a rule group or injection group in intermittent mode, the conversion sequence does not automatically start from the beginning when it ends. When all subgroups have been converted, the next trigger event starts the conversion of the first subgroup.*

*2. You cannot use auto-injection (IAUTO=1) and intermittent mode at the same time.*

*3. You cannot set intermittent mode for both rule groups and injection groups, and intermittent mode can only be used for a group of conversions.*

**4) Continuous conversion**

In the continuous conversion mode, another conversion is started as soon as the previous ADC conversion is finished, and the conversion does not stop on the last channel of the selection group, but continues again from the first channel of the selection group. The startup events for this mode include an external trigger event and ADON position 1, which is required to set the CONT position 1 after startup.

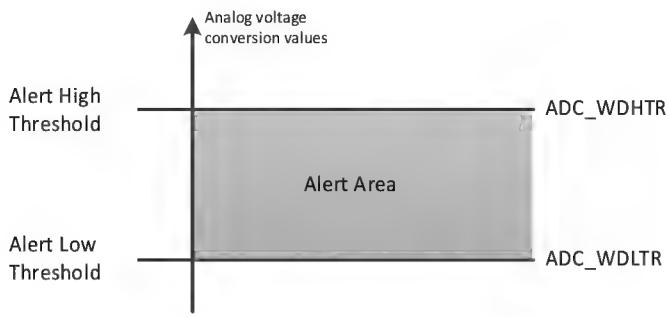
If a rule channel is converted, the conversion data is stored in the ADC_RDATAR register, and the end-of-conversion flag EOC is set, generating an interrupt if EOCIE is set.

If an injection channel is converted, the conversion data is stored in the ADC_IDATARx register, the end-of-injection-conversion flag JEOC is set, and an interrupt is generated if JEOCIE is set.

### 10.2.5 Analog Watchdog

The AWD analog watchdog status bit is set if the analog voltage being converted by the ADC is below the low threshold or above the high threshold. The threshold settings are located in the lowest 12 valid bits of the ADC_WDHTR and ADC_WDLTR registers. The AWDIE bit of the ADC_CTLR1 register is set to allow the corresponding interrupt to be generated.

Figure 10-4 Analog watchdog threshold area




Configure the AWDSGL, AWDEN, JAWDEN and AWDCH[3:0] bits of the ADC_CTLR1 register to select the

channel for analog watchdog alerting, as related in the following table.

Table 10-4 Analog Watchdog channel selection

| Analog Watchdog alert channel | ADC_CTLR1 register control bit | | | |
|---|---|---|---|---|
| | AWDSGL | AWDEN | JAWDEN | AWDCH[3:0] |
| No vigilance | Ignore | 0 | 0 | Ignore |
| All injection channels | 0 | 0 | 1 | Ignore |
| All rule channels | 0 | 1 | 0 | Ignore |
| All injection and rule channels | 0 | 1 | 1 | Ignore |
| Single injection channel | 1 | 0 | 1 | Determine the channel number |
| Single rule channel | 1 | 1 | 0 | Determine the channel number |
| Single injection and rule channel | 1 | 1 | 1 | Determine the channel number |

## 10.3 Register Description

Table 10-5 ADC-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_ADC_STATR | 0x40012400 | ADC status register | 0x00000000 |
| R32_ADC_CTLR1 | 0x40012404 | ADC control register 1 | 0x00000000 |
| R32_ADC_CTLR2 | 0x40012408 | ADC control register 2 | 0x00000000 |
| R32_ADC_SAMPTR1 | 0x4001240C | ADC sample time configuration register 1 | 0x00000000 |
| R32_ADC_SAMPTR2 | 0x40012410 | ADC sample time configuration register 2 | 0x00000000 |
| R32_ADC_IOFR1 | 0x40012414 | ADC injected channel data offset register 1 | 0x00000000 |
| R32_ADC_IOFR2 | 0x40012418 | ADC injected channel data offset register 2 | 0x00000000 |
| R32_ADC_IOFR3 | 0x4001241C | ADC injected channel data offset register 3 | 0x00000000 |
| R32_ADC_IOFR4 | 0x40012420 | ADC injected channel data offset register 4 | 0x00000000 |
| R32_ADC_WDHTR | 0x40012424 | ADC watchdog high threshold register | 0x00000FFF |
| R32_ADC_WDLTR | 0x40012428 | ADC watchdog low threshold register | 0x00000000 |
| R32_ADC_RSQR1 | 0x4001242C | ADC regular sequence register 1 | 0x00000000 |
| R32_ADC_RSQR2 | 0x40012430 | ADC regular sequence register 2 | 0x00000000 |
| R32_ADC_RSQR3 | 0x40012434 | ADC regular sequence register 3 | 0x00000000 |
| R32_ADC_ISQR | 0x40012438 | ADC injected sequence register | 0x00000000 |
| R32_ADC_IDATAR1 | 0x4001243C | ADC injected data register 1 | 0x00000000 |
| R32_ADC_IDATAR2 | 0x40012440 | ADC injected data register 2 | 0x00000000 |
| R32_ADC_IDATAR3 | 0x40012444 | ADC injected data register 3 | 0x00000000 |
| R32_ADC_IDATAR4 | 0x40012448 | ADC injected data register 4 | 0x00000000 |
| R32_ADC_RDATAR | 0x4001244C | ADC regular data register | 0x00000000 |
| R32_ADC_CTLR3 | 0x40012450 | ADC control register 3 | 0x00000003 |
| R32_ADC_WDTR1 | 0x40012454 | ADC watchdog threshold 1 register | 0x0FFF0000 |
| R32_ADC_WDTR2 | 0x40012458 | ADC watchdog threshold 2 register | 0x0FFF0000 |
| R32_ADC_WDTR3 | 0x4001245C | ADC watchdog threshold 3 register | 0x0FFF0000 |

### 10.3.1 ADC Status Register (ADC_STATR)

Offset address: 0x00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | STRT | JSTRT | JEOC | EOC | AWD |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:5] | Reserved | RO | Reserved. | 0 |
| 4 | STRT | RW0 | Rule channel transition start state.<br>1: Rule channel conversion has started.<br>0: Rule channel conversion is not started.<br>This bit is set to 1 by hardware and cleared to 0 by software (Write 1 is not valid). | 0 |
| 3 | JSTRT | RW0 | Injection channel conversion start state.<br>1: Injection channel conversion has started.<br>0: Injection channel conversion has not started.<br>This bit is set to 1 by hardware and cleared to 0 by software (Write 1 is not valid). | 0 |
| 2 | JEOC | RW0 | Injection into the end state of the channel group conversion.<br>1: Conversion complete.<br>0: The conversion is not completed.<br>This bit is set to 1 by hardware (All injected channels are converted) and cleared to 0 by software (Write 1 is invalid). | 0 |
| 1 | EOC | RW0 | Conversion end state.<br>1: Conversion complete.<br>0: The conversion is not completed.<br>This bit is set to 1 by hardware (End of rule or injection channel group conversion), cleared by software to 0 (Write 1 is invalid) or when reading ADC_RDATAR. | 0 |
| 0 | AWD | RW0 | Analog watchdog flag bit.<br>1: Occurrence of simulated watchdog events.<br>0: No simulated watchdog event occurred.<br>This bit is set to 1 by hardware (Conversion value is out of range of ADC_WDHTR and ADC_WDLTR registers) and cleared to 0 by software (Write 1 is not valid). | 0 |

### 10.3.2 ADC Control Register 1 (ADC_CTLR1)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | TKENABLE | AWDEN | JAWDEN | Reserved | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DISCNUM[2:0] | | | JDISCEN | DISCEN | JAUTO | AWDSGL | SCAN | JEOCIE | AWDIE | EOCIE | Reserved | AWDCH[3:0] | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:25] | Reserved | RO | Reserved. | 0 |
| 24 | TKENABLE | RW | TKEY module enable control, including TKEY_F and TKEY_V units:<br>1: Enable the TKEY module; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: Disable the TKEY module. | |
| 23 | AWDEN | RW | Analog watchdog function enable bit on the rule channel.<br>1: Enable the analog watchdog on the rule channel.<br>0: Disable the analog watchdog on the rule channel. | 0 |
| 22 | JAWDEN | RW | Analog watchdog function enable bit on the injection channel.<br>1: Enable the analog watchdog on the injection channel.<br>0: Disable the analog watchdog on the injection channel. | 0 |
| [21:16] | Reserved | RO | Reserved. | 0 |
| [15:13] | DISCNUM[2:0] | RW | Number of rule channels to be converted after external triggering in intermittent mode.<br>000: 1 channel.<br>...<br>111: 8 channels. | 0 |
| 12 | JDISCEN | RW | Inject the intermittent mode enable bit on the channel.<br>1: Enable intermittent mode on the injection channel.<br>0: Disable intermittent mode on the injection channel. | 0 |
| 11 | DISCEN | RW | Intermittent mode enable bit on rule channel.<br>1: Enable intermittent mode on the rule channel.<br>0: Disable intermittent mode on the rule channel. | 0 |
| 10 | JAUTO | RW | After the opening of the rule channel is completed, the injection channel group enable bit is automatically switched.<br>1: Enable automatic injection channel group switching.<br>0: Disable automatic injection channel group conversion.<br>*Note: This mode requires disabling the external trigger function of the injection channel.* | 0 |
| 9 | AWDSGL | RW | In scan mode, use the analog watchdog enable bit on a single channel.<br>1: Use an analog watchdog on a single channel (AWDCH[4:0] selection).<br>0: Use analog watchdog on all channels. | 0 |
| 8 | SCAN | RW | Scan mode enable bit.<br>1: Enable scan mode (Continuous conversion of all channels selected by ADC_IOFRx and ADC_RSQRx).<br>0: Disable scan mode. | 0 |
| 7 | JEOCIE | RW | Inject the channel group end-of-conversion interrupt enable bit.<br>1: Enable the injection of the channel group conversion completion interrupt (JEOC flag).<br>0: Disable the injection channel group conversion completion interrupt. | 0 |
| 6 | AWDIE | RW | Analog watchdog interrupt enable bit.<br>1: Enable the analog watchdog interrupt.<br>0: Disable the analog watchdog interrupt.<br>*Note: In scan mode, this interrupt will abort the scan if it occurs.* | 0 |
| 5 | EOCIE | RW | End of conversion (Rule or injection channel group) interrupt enable bit.<br>1: Enable the end-of-conversion interrupt (EOC flag).<br>0: Disable the end-of-conversion interrupt. | 0 |
| 4 | Reserved | RO | Reserved. | 0 |
| [3:0] | AWDCH[3:0] | RW | Analog watchdog channel selection bits.<br>00000: Analog input channel 0.<br>00001: Analog input channel 1.<br>...<br>01111: Analog input channel 15. | 0 |

### 10.3.3 ADC Control Register 2 (ADC_CTLR2)

Offset address: 0x08

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | SWSTART | JSWSTART | EXTTRIG | EXTSEL[2:0] | | | Reserved |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| JEXTTRIG | JEXTSEL[2:0] | | | ALIGN | Reserved | | DMA | Reserved | | | | | | CONT | ADON |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:23] | Reserved | RO | Reserved. | 0 |
| 22 | SWSTART | RW | To start a rule channel conversion, you need to set the software trigger to.<br>1: Initiate rule channel conversion.<br>0: Reset state.<br>This bit is set by software and cleared to 0 by hardware when conversion starts. | 0 |
| 21 | JSWSTART | RW | To initiate an injection channel transition, set the software to trigger:<br>1: Initiates an injection channel transition;<br>0: Reset state.<br>This bit is set by software and is cleared to 0 by hardware or 0 by software when the conversion starts. | 0 |
| 20 | EXTTRIG | RW | Externally triggered conversion mode enable for rule channels:<br>1: Initiation of the transition using an external event;<br>0: External event start function disabled. | 0 |
| [19:17] | EXTSEL[2:0] | RW | External trigger event selection for initiating rule channel conversion.<br>000: TRGO event of timer 1;<br>001: CC1 event of timer 1;<br>010: CC2 event of timer 1;<br>011: TRGO event of timer 2;<br>100: CC1 event of timer 2;<br>101: CC1 event of timer 3;<br>110: EXTI line 11;<br>111: SWSTART software trigger. | 0 |
| 16 | Reserved | RO | Reserved. | 0 |
| 15 | JEXTTRIG | RW | Externally triggered conversion mode enable for the injected channels:<br>1: initiation of the transition using an external event;<br>0: external event start function disabled. | 0 |
| [14:12] | JEXTSEL[2:0] | RW | Selection of external trigger events to initiate the injection of channel transitions:<br>000: CC3 event of timer 1;<br>001: CC4 event of timer 1;<br>010: CC3 event of timer 2;<br>011: CC4 event of timer 2;<br>100: CC2 event of timer 2;<br>101: CC2 event of timer 3;<br>110: EXTI line 15; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 111: JSWSTART software trigger. | |
| 11 | ALIGN | RW | Data alignment.<br>1: Left-aligned; 0: Right-aligned.<br>*Note: Data left-aligned applies only when rule groups are used.* | 0 |
| [10:9] | Reserved | RO | Reserved. | 0 |
| 8 | DMA | RW | Direct Memory Access (DMA) mode enable.<br>1: Enable DMA mode.<br>0: Disable DMA mode. | 0 |
| [7:2] | Reserved | RO | Reserved. | 0 |
| 1 | CONT | RW | Continuous conversion enable:<br>1: Continuous conversion mode;<br>0: Single conversion mode.<br>If this bit is set, conversion will be continuous until the bit is cleared. | 0 |
| 0 | ADON | RW | On/off A/D converter<br>When this bit is 0, writing 1 will wake up the ADC from power-down mode; when this bit is 1, writing 1 will start the conversion.<br>1: Turn on the ADC and start the conversion.<br>0: Turn off ADC conversion and enter power-down mode.<br>*Note: A conversion is initiated when only ADON is changed in the register, and no new conversion is initiated if there are any other bits sent for change.* | 0 |

### 10.3.4 ADC Sample Time Configuration Register 1 (ADC_SAMPTR1)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | SMP17[2:0] | | | SMP16[2:0] | | | SMP15[2:1] | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SMP15[0] | SMP14[2:0] | | | SMP13[2:0] | | | SMP12[2:0] | | | SMP11[2:0] | | | SMP10[2:0] | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved. | 0 |
| [23:0] | SMPx[2:0] | RW | SMPx[2:0]: sampling time configuration for channel x (a-f):<br>000: 4 cycles; 001: 5 cycles;<br>010: 6 cycles; 011: 7 cycles;<br>100: 8 cycles; 101: 9 cycles;<br>110: 10 cycles; 111: 11 cycles;<br>These bits are used to select the sampling time for each channel independently, and the channel configuration value must remain constant during the sampling cycle. | 0 |

### 10.3.5 ADC Sample Time Configuration Register 2 (ADC_SAMPTR2)

Offset address: 0x10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | SMP9[2:0] | | | SMP8[2:0] | | | SMP7[2:0] | | | SMP6[2:0] | | | SMP5[2:1] | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

| SMP5[0] | SMP4[2:0] | SMP3[2:0] | SMP2[2:0] | SMP1[2:0] | SMP0[2:0] |
|---|---|---|---|---|---|

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved. | 0 |
| [29:0] | SMPx[2:0] | RW | SMPx[2:0]: sampling time configuration for channel x (0-9):<br>000: 4 cycles; 001: 5 cycles;<br>010: 6 cycles; 011: 7 cycles;<br>100: 8 cycles; 101: 9 cycles;<br>110: 10 cycles; 111: 11 cycles;<br>These bits are used to select the sampling time for each channel independently, and the channel configuration value must remain constant during the sampling cycle. | 0 |

### 10.3.6 ADC Injected Channel Data Offset Register x (ADC_IOFRx) (x=1/2/3/4)

Offset address: 0x14 + (x-1)*4

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | JOFFSETx[11:0] | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | Reserved | RO | Reserved. | 0 |
| [11:0] | JOFFSETx | RW | The data offset value of the injected channel x.<br>When converting the injected channels, this value defines the value used to subtract from the original conversion data. The result of the conversion can be read out in the ADC_IDATARx register. | 0 |

### 10.3.7 ADC Watchdog High Threshold Register (ADC_WDHTR)

Offset address: 0x24

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | HT[11:0] | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | Reserved | RO | Reserved. | 0 |
| [11:0] | HT[11:0] | RW | Analog watchdog high threshold setting value. | 0xfff |

*Note: You can change the values of WDHTR and WDLTR during the conversion process, but they will take effect at the next conversion.*

### 10.3.8 ADC Watchdog Low Threshold Register (ADC_WDLTR)

Offset address: 0x28

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

| Reserved | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | LT[11:0] | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | Reserved | RO | Reserved. | 0 |
| [11:0] | LT[11:0] | RW | Analog watchdog low threshold setting value. | 0 |

*Note: You can change the values of WDHTR and WDLTR during the conversion process, but they will take effect at the next conversion.*

### 10.3.9 ADC Regular Sequence Register 1(ADC_RSQR1)

Offset address: 0x2C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | L[3:0] | | | | SQ16[4:1] | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SQ16[0] | SQ15[4:0] | | | | | SQ14[4:0] | | | | | SQ13[4:0] | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved. | 0 |
| [23:20] | L[3:0] | RW | Number of channels to be converted in a regular channel conversion sequence.<br>0000-1111: 1-16 conversions. | 0 |
| [19:15] | SQ16[4:0] | RW | The number of the 16th conversion channel in the rule sequence (0-13, 15). | 0 |
| [14:10] | SQ15[4:0] | RW | The number of the 15th conversion channel in the rule sequence (0-13, 15). | 0 |
| [9:5] | SQ14[4:0] | RW | The number of the 14th conversion channel in the rule sequence (0-13, 15). | 0 |
| [4:0] | SQ13[4:0] | RW | The number of the 13th conversion channel in the rule sequence (0-13, 15). | 0 |

### 10.3.10 ADC Regular Sequence Register 2 (ADC_RSQR2)

Offset address: 0x30

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | SQ12[4:0] | | | | | SQ11[4:0] | | | | | SQ10[4:1] | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SQ10[0] | SQ9[4:0] | | | | | SQ8[4:0] | | | | | SQ7[4:0] | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved. | 0 |
| [29:25] | SQ12[4:0] | RW | The number of the 12th conversion channel in the rule sequence (0-13, 15). | 0 |
| [24:20] | SQ11[4:0] | RW | The number of the 11th conversion channel in the rule sequence (0-13, 15). | 0 |
| [19:15] | SQ10[4:0] | RW | The number of the 10th conversion channel in the rule sequence (0-13, 15). | 0 |

| | | | | |
|---|---|---|---|---|
| [14:10] | SQ9[4:0] | RW | The number of the 9th conversion channel in the rule sequence (0-13, 15). | 0 |
| [9:5] | SQ8[4:0] | RW | The number of the 8th conversion channel in the rule sequence (0-13, 15). | 0 |
| [4:0] | SQ7[4:0] | RW | The number of the 7th conversion channel in the rule sequence (0-13, 15). | 0 |

### 10.3.11 ADC Regular Sequence Register 3 (ADC_RSQR3)

Offset address: 0x34

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | SQ6[4:0] | | | | | SQ5[4:0] | | | | | SQ4[4:1] | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SQ4[0] | | SQ3[4:0] | | | | | SQ2[4:0] | | | | | SQ1[4:0] | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved. | 0 |
| [29:25] | SQ6[4:0] | RW | The number of the 6th conversion channel in the rule sequence (0-13, 15). | 0 |
| [24:20] | SQ5[4:0] | RW | The number of the 5th conversion channel in the rule sequence (0-13, 15). | 0 |
| [19:15] | SQ4[4:0] | RW | The number of the 4th conversion channel in the rule sequence (0-13, 15). | 0 |
| [14:10] | SQ3[4:0] | RW | The number of the 3th conversion channel in the rule sequence (0-13, 15). | 0 |
| [9:5] | SQ2[4:0] | RW | The number of the 2th conversion channel in the rule sequence (0-13, 15). | 0 |
| [4:0] | SQ1[4:0] | RW | The number of the 1th conversion channel in the rule sequence (0-13, 15). | 0 |

### 10.3.12 ADC Injected Sequence Register (ADC_ISQR)

Offset address: 0x38

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | JL[1:0] | | JSQ4[4:1] | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| JSQ4[0] | | JSQ3[4:0] | | | | | JSQ2[4:0] | | | | | JSQ1[4:0] | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:22] | Reserved | RO | Reserved. | 0 |
| [21:20] | JL[1:0] | RW | Inject the number of channels to be converted in the channel conversion sequence.<br>00-11: 1-4 conversions. | 0 |
| [19:15] | JSQ4[4:0] | RW | The number of the 4th conversion channel in the injection sequence (0-13, 15).<br>*Note: The software writes and assigns the channel number (0-13, 15) as the 4th in the sequence to be converted.* | 0 |
| [14:10] | JSQ3[4:0] | RW | The number of the 3th conversion channel in the injection sequence (0-13, 15). | 0 |
| [9:5] | JSQ2[4:0] | RW | The number of the 2th conversion channel in the injection | 0 |

| | | | | |
|---|---|---|---|---|
| | | | sequence (0-13, 15). | |
| [4:0] | JSQ1[4:0] | RW | The number of the 1th conversion channel in the injection sequence (0-13, 15). | 0 |

*Note: Unlike the regular conversion sequence, if the length of JL[1:0] is less than 4, the sequence order of conversion starts from (4 - JL).*

*For example, when JL[1:0]=3 (4 injected transitions in the sequencer), the ADC will convert channels in the following order: JSQ1[4:0], JSQ2[4:0], JSQ3[4:0], and JSQ4[4:0];*

*When JL[1:0]=2 (3 injected transitions in the sequencer), the ADC will convert the channels in the following order: JSQ2[4:0], JSQ3[4:0] and JSQ4[4:0];*

*When JL[1:0]=1 (2 injected conversions in the sequencer), the ADC converts the channels in the following order: first JSQ3[4:0], then JSQ4[4:0];*

*When JL[1:0] = 0 (1 injection conversion in the sequencer), the ADC will convert only the JSQ4[4:0] channels.*

*If ADCx_ISQR[21:0]=10 00111 00011 00111 00010, the ADC will convert channels in the following order: JSQ2[4:0], JSQ3[4:0], and JSQ4[4:0], indicating that the scan conversions are performed in the following channel order: 7, 3, 7.*

### 10.3.13 ADC Injected Data Register (ADC_IDATARx) (x=1/2/3/4)

Offset address: 0x3C + (x-1)*4

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| JDATA[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved. | 0 |
| [15:0] | JDATA[15:0] | RO | Injection of channel conversion data (Data right-aligned). | 0 |

### 10.3.14 ADC Regular Data Register (ADC_RDATAR)

Offset address: 0x4C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DATA[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved. | 0 |
| [15:0] | DATA[15:0] | RO | Rule channel conversion data (Data left-aligned or right-aligned) | 0 |

### 10.3.15 ADC Control Register 3 (ADC_CTLR3)

Offset address: 0x50

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | AWD | AWD | AWD | AWD |

| | | | | | | | | | | | | 3_RES | 2_RES | 1_RES | 0_RES |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| AED_RST_EN[3:0] | | | | Reserved | | AWD_SCAN | Reserved | | | | | CLK_DIV[3:0] | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:20] | Reserved | RO | Reserved. | 0 |
| 19 | AWD3_RES | RW0 | Analog watchdog 3 comparison results.<br>1: The conversion value is greater than the Watchdog 3 high threshold or less than the Watchdog 3 low threshold;<br>0: The conversion value is between the Watchdog 3 high threshold and low threshold.<br>*Note: Hardware set to 1, write 0 to clear.* | 0 |
| 18 | AWD2_RES | RW0 | Analog watchdog 2 comparison results.<br>1: The conversion value is greater than the Watchdog 2 high threshold or less than the Watchdog 2 low threshold;<br>0: The conversion value is between the Watchdog 2 high threshold and low threshold.<br>*Note: Hardware set to 1, write 0 to clear.* | |
| 17 | AWD1_RES | RW0 | Analog watchdog 1 comparison results.<br>1: The conversion value is greater than the Watchdog 1 high threshold or less than the Watchdog 1 low threshold;<br>0: The conversion value is between the Watchdog 1 high threshold and low threshold.<br>*Note: Hardware set to 1, write 0 to clear.* | |
| 16 | AWD0_RES | RW0 | Analog watchdog 0 comparison results.<br>1: The conversion value is greater than the Watchdog 0 high threshold or less than the Watchdog 0 low threshold;<br>0: The conversion value is between the Watchdog 0 high threshold and low threshold.<br>*Note: Hardware set to 1, write 0 to clear.* | |
| 15 | AWD3_RST_EN | RW | Watchdog 3 output reset enable bit:<br>1: On;<br>0: Off. | 0 |
| 14 | AWD2_RST_EN | RW | Watchdog 2 output reset enable bit:<br>1: On;<br>0: Off. | 0 |
| 13 | AWD1_RST_EN | RW | Watchdog 1 output reset enable bit:<br>1: On;<br>0: Off. | 0 |
| 12 | AWD0_RST_EN | RW | Watchdog 0 output reset enable bit:<br>1: On;<br>0: Off. | 0 |
| [11:10] | Reserved | RO | Reserved. | 0 |
| 9 | AWD_SCAN | RW | Analog watchdog scan enable:<br>1: Enable watchdog scanning;<br>0: Disable watchdog scanning. | 0 |
| [8:4] | Reserved | RO | Reserved. | 0 |
| [3:0] | CLK_DIV[3:0] | RW | Frequency division factor:<br>0000: Prescaler off; 0001: Divided by 2;<br>0010: Divided by 3; 0011: Divided by 4;<br>0100: Divided by 5; 0101: Divided by 6; | 0011b |

| | | | | |
|---|---|---|---|---|
| | | | 0110: Divided by 7; 0111: Divided by 8;<br>1000: Divided by 9; 1001: Divided by 10;<br>1010: Divided by 11; 1011: Divided by 12;<br>1100: Divided by 13; 1101: Divided by 14;<br>1110: Divided by 15; 1111: Divided by 16;<br>*Note: A crossover factor greater than 5 is recommended and the ADC clock frequency is recommended to be less than 1/2 the system clock.* | |

### 10.3.16 ADC Watchdog 1 Threshold Register (ADC_WDTR1)

Offset address: 0x54

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | HTR1 | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | LTR1 | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:28] | Reserved | RO | Reserved | 0 |
| [27:16] | HTR1 | RW | Analog watchdog high threshold setting value. | 0xFFF |
| [15:12] | Reserved | RO | Reserved | 0 |
| [11:0] | LTR1 | RW | Analog watchdog low threshold setting value. | 0x000 |

*Note: Only available for watchdog channel 1.*

### 10.3.17 ADC Watchdog 2 Threshold Register (ADC_WDTR2)

Offset address: 0x58

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | HTR2 | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | LTR2 | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:28] | Reserved | RO | Reserved. | 0 |
| [27:16] | HTR2 | RW | Analog watchdog high threshold setting value. | 0xFFF |
| [15:12] | Reserved | RO | Reserved | 0 |
| [11:0] | LTR2 | RW | Analog watchdog low threshold setting value. | 0x000 |

*Note: Only available for watchdog channel 2.*

### 10.3.18 ADC Watchdog 3 Threshold Register (ADC_WDTR3)

Offset address: 0x5C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | HTR3 | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | LTR3 | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved. | 0 |
| [27:16] | HTR3 | RW | Analog watchdog high threshold setting value. | 0xFFF |
| [15:12] | Reserved | RO | Reserved | 0 |
| [11:0] | LTR3 | RW | Analog watchdog low threshold setting value. | 0x000 |

*Note: Only available for watchdog channel 3.*

# Chapter 11 Touch Key Detection (TKEY)

The touch detection control (TKEY) unit, with the aid of the ADC module's voltage conversion function, implements the touch button detection function by converting the electrical capacity into a voltage quantity for sampling. The detection channels are alternate with the 14 external channels of the ADC and the touch button detection is achieved by the single conversion mode of the ADC module.

## 11.1 TKEY Functional Description

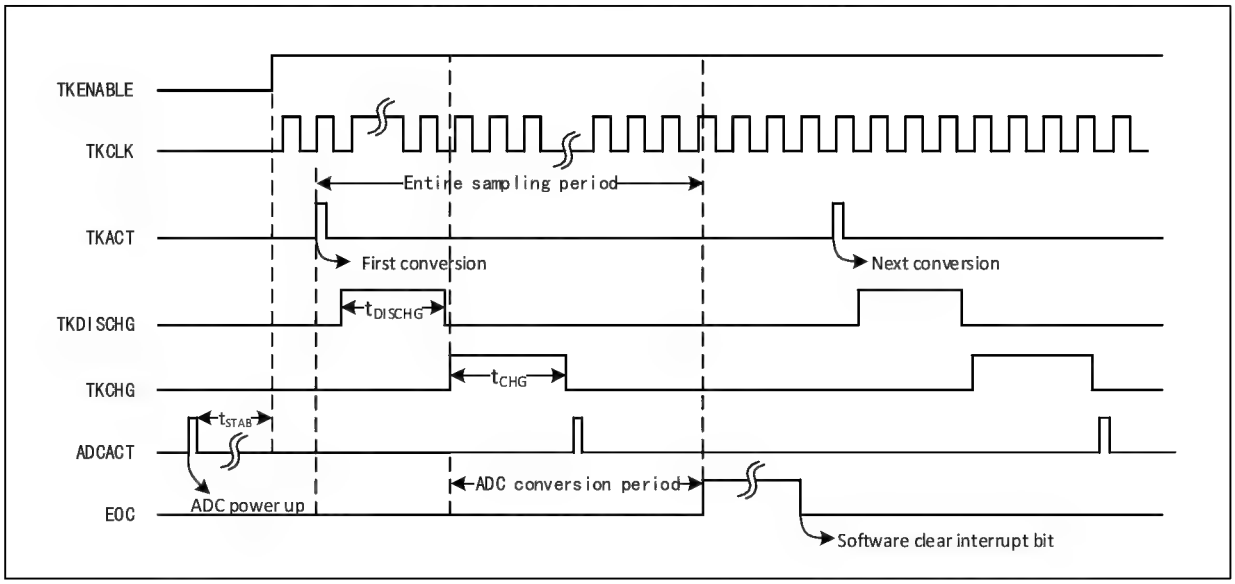
- TKEY on

The TKEY detection process requires the cooperation of the ADC module to carry out, so when using the TKEY function, you need to ensure that the ADC module is in the power-on state (ADON=1), then turn on the TKEY unit function by setting the TKENABLE position 1 of the ADC_CTLR1 register.

TKEY only supports single channel conversion mode, configure the channel to be converted to the first of the ADC module's rule group sequence and the software initiates the conversion (Write the TKEY_ACT_DCG register).

*Note: When TKEY conversion is not performed, the ADC channel configuration conversion function can still be retained.*

Figure 11-1 TKEY operating timing diagram




- Programmable sampling times

The TKEY cell conversion requires a discharge using a number of ADCCLK clock cycles ($t_{DISCHG}$) followed by a voltage sample by charging the channel over a number of ADCCLK cycles ($t_{CHG}$). The number of charging cycles is the sum of the TKCGx[2:0] configuration values in the TKEY_CHARGE1 and TKEY_CHARGE2 registers plus The sum of the TKEY_CHGOFFSET offsets, each channel can be adjusted separately with a different charge cycle to sample the voltage.

## 11.2 TKEY Operation Procedures

The TKEY detection is an extension of the ADC module and works by converting the capacity sensed by the hardware channel through "touch" and "non-touch" methods, and then converting the change in capacity to a voltage through a configurable number of charge/discharge cycles. This is converted into a digital value by the ADC module.

The ADC is configured to operate in a single pass single channel mode during sampling, with the TKEY_ACT register "write operation" initiating a conversion as follows:

1) Initialize the ADC function, configure the ADC module as a single conversion module, set the ACON bit to 1 and wake up the ADC module. Set the TKENABLE position of ADC_CTLR1 register to 1 and open the TKEY unit.

2) Set the channel to be converted, write the channel number to the first conversion position (ADC_RSQR3[4:0]) in the ADC rule group sequence and set L[3:0] to 1.

3) Set the charging sample time for the channel, write the TKEY_CHARGEx register to configure a different charging time for each channel.

4) Write the TKEY_CHGOFFSET register to set the charge time offset of the channel (Valid in the lower eight bits) to adjust the charge time.

5) Write the TKEY_ACT_DCG register to set the discharge time (Valid in the lower eight bits) and to initiate a sample and conversion of the TKEY.

6) Wait for the EOC end of conversion flag position 1 in the ADC status register and read the ADC_DR register to get the value of this conversion.

7) If the next conversion is required, repeat steps 2-6. If the channel charge sampling time does not need to be modified, step 3 or 4 can be omitted.

## 11.3 TKEY Register Description

Table 11-1 TKEY1-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_TKEY1_CHARGE1 | 0x4001240C | TKEY charge sample time register 1 | 0x00000000 |
| R32_TKEY1_CHARGE2 | 0x40012410 | TKEY charge sample time register 2 | 0x00000000 |
| R32_TKEY1_CHGOFFSET | 0x4001243C | TKEY charge time offset register | 0x00000000 |
| R32_TKEY1_ACT_DCG | 0x4001244C | TKEY start and discharge time register | 0x00000000 |
| R32_TKEY1_DR | 0x4001244C | TKEY data register | 0x00000000 |

### 11.3.1 TKEYx Charge Sample Time Register 1 (TKEYx_CHARGE1) (x=1)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | TKCG17[2:0] | | | TKCG16[2:0] | | | TKCG15[2:1] | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TKCG15 | TKCG14[2:0] | | | TKCG13[2:0] | | | TKCG12[2:0] | | | TKCG11[2:0] | | | TKCG10[2:0] | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved. | 0 |
| [23:0] | TKCGx[2:0] | RW | TKCGx[2:0] (x=10-17): Selects the charge sample time for channel x.<br>These are used to select the charge times for each channel independently.<br>000: 4 cycles 100: 5 cycles<br>001: 6 cycles 101: 7 cycles<br>010: 8 cycles 110: 9 cycles<br>011: 10 cycles 111: 11 cycles<br>Time reference: ADC clock. | 0 |

*Note: This register maps the ADC module's sample time register 1 (ADC_SAMPTR1). When configuring the ADC function, it is the adoption time of the channel; when configuring the TKEY function, it is the charging time of the channel.*

### 11.3.2 TKEYx Charge Sample Time Register 2 (TKEYx_CHARGE2) (x=1)

Offset address: 0x10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | TKCG9[2:0] | | | TKCG8[2:0] | | | TKCG7[2:0] | | | TKCG6[2:0] | | | TKCG5[2:1] | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TKCG5 | | TKCG4[2:0] | | | TKCG3[2:0] | | | TKCG2[2:0] | | | TKCG1[2:0] | | | TKCG0[2:0] | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:30] | Reserved | RO | Reserved. | 0 |
| [29:0] | TKCGx[2:0] | RW | TKCGx[2:0] (x=0-9): Selects the charge sample time for channel x.<br>These are used to select the charge times for each channel independently.<br>000: 4 cycles 100: 5 cycles<br>001: 6 cycles 101: 7 cycles<br>010: 8 cycles 110: 9 cycles<br>011: 10 cycles 111: 11 cycles<br>Time reference: ADC clock. | 0 |

*Note: This register maps the ADC module's sample time register 2 (ADC_SAMPTR2). When configuring the ADC function, it is the adoption time of the channel; when configuring the TKEY function, it is the charging time of the channel.*

### 11.3.3 TKEYx Charge Time Offset Register (TKEYx_CHGOFFSET) (x=1)

Offset address: 0x3C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | TKCGOFFSET[7:0] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | TKCGOFFSET[7:0] | WO | TKEY charge time offset configuration | 0 |

| | | | | |
|---|---|---|---|---|
| | | | value.<br>Total charge time TCHG=TKCGOFFSET (Unit: system clock cycle) | |

*Note: This register maps the ADC module's injection data register 1 (ADC_IDATAR1). Therefore, when this address register is used for a "write operation", it is executed as the TKEY charge time offset (TKEY_CHGOFFSET); when it is used for a "read operation", it is executed as the ADC module's injection data register 1 (ADC_ IDATAR1).*

### 11.3.4 TKEYx Start and Discharge Time Register (TKEYx_ACT_DCG) (x=1)

Offset address: 0x4C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | | TKACT_DCG[7:0] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | TKACT_DCG[7:0] | WO | Write the discharge time and initiate a TKEY channel detection. | 0 |

*Note: This register maps the ADC module's rule data register (ADC_RDATAR).*

### 11.3.5 TKEYx Data Register (TKEYx_DR) (x=1)

Offset address: 0x4C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| DATA[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0 |
| [15:0] | DATA[15:0] | RO | Converted data. | 0 |

*Note: This register maps the ADC module's rule data register (ADC_RDATAR).*

# Chapter 12 Advanced-control Timer (ADTM)

The Advanced-control timer Module contains 2 powerful 16-bit auto-reload timers (TIM1/TIM2), which can be used to measure pulse width or generate pulses, PWM waves, etc. It is used in motor control, power supply, etc.

## 12.1 Main Features

The main features of the advanced-control timer (TIM1/TIM2) include.

- 16-bit auto-reload counter supporting incremental counting mode, decremental counting mode and incremental and decremental counting mode.
- 16-bit prescaler with dynamically adjustable crossover coefficients from 1 to 65536.
- Support four independent comparison capture channels.
- Each comparison capture channel supports multiple operating modes, such as: input capture, output comparison, PWM generation and single pulse output.
- Support complementary outputs with programmable dead time;
- Support external signals to control the timer.
- Support updating the timer after a defined period using a repeat counter.
- Support resetting the timer or placing it in the OK state using the brake signal.
- Support the use of DMA in multiple modes.
- Support incremental encoders.
- Support cascading and synchronization between timers.

## 12.2 Principle and Structure

This section deals with the internal construction of advanced-control timers.

### 12.2.1 Overview

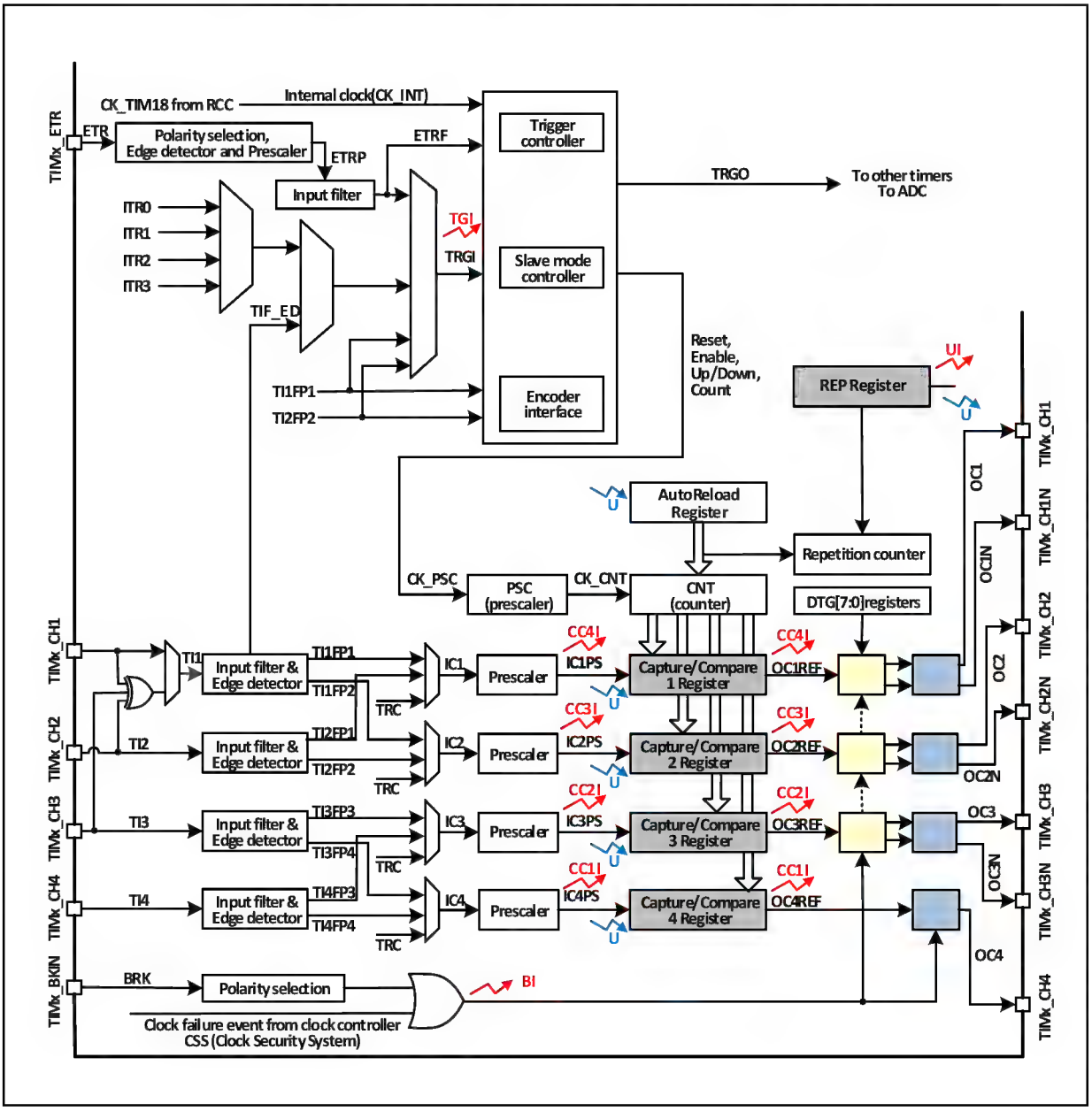
As shown in Figure 12-1, the structure of the advanced-control timer can be roughly divided into three parts, namely the input clock part, the core counter part and the compare capture channel part.

The advanced-control timer can be clocked from the HB bus clock (CK_INT), from an external clock input pin (TIMx_ETR), from other timers with clock output (ITRx), or from the input of the compare capture channel (TIMx_CHx). These input clock signals become the CK_PSC clock after various set filtering and dividing operations and are output to the core counter section. In addition, these complex clock sources can also be output as TRGO to other peripherals such as timers and ADCs.

The core of the advanced-control timer is a 16-bit counter (CNT), and the CK_PSC is divided by a prescaler (PSC) to become the CK_CNT and output to the CNT. An auxiliary counter counts the number of times the ATRLR reloads the initial value for the CNT and generates a specific event when the count reaches the number set in the Repeat Count Register (RPTCR).

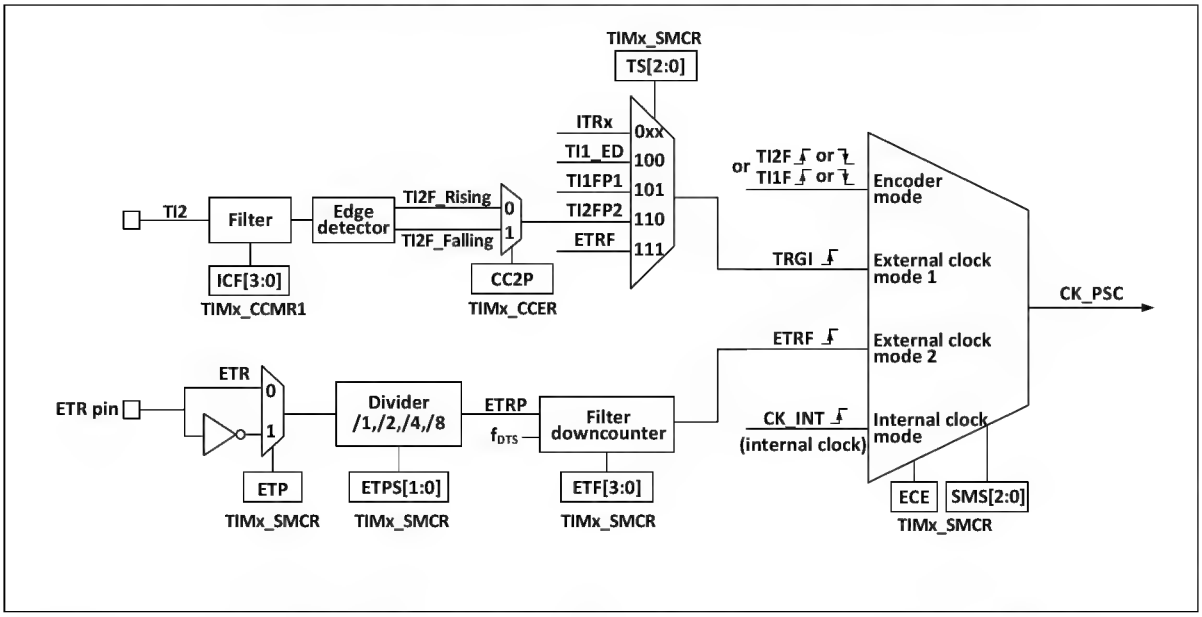
The advanced-control timer has four sets of compare capture channels, each of which can input pulses from exclusive pins or output waveforms to the pins, i.e., the compare capture channels support both input and output modes. The input of each channel of the compare capture register supports filtering, dividing and edge detection operations, and supports mutual triggering between channels, as well as providing a clock for the core counter CNT. Each compare capture channel has a set of compare capture registers (CHxCVR) that support comparison with the main counter (CNT) to output pulses.

Figure 12-1 Block diagram of advanced-control timer structure



## 12.2.2 Clock Input

Figure 12-2 Block diagram of CK_PSC source for advanced-control timer




The advanced-control timer CK_PSC has many clock sources and can be divided into 4 categories.

1) Route of external clock pin (ETR) input clock: ETR → ETRP → ETRF.
2) Internal HB clock input route: CK_INT.
3) Route from the comparison capture channel pin (TIMx_CHx): TIMx_CHx → TIx → TIxFPx, this route is also used in encoder mode.
4) Inputs from other internal timers: ITRx.

The actual operation can be divided into 4 categories by determining the choice of input pulse for the SMS of the CK_PSC source.

1) Selection of the internal clock source (CK_INT).
2) External clock source mode 1.
3) External clock source mode 2.
4) Encoder mode.

All 4 clock source sources mentioned above can be selected by these 4 operations.

### 12.2.2.1 Internal Clock Source (CK_INT)

If the SMS field is held at 000b to start the advanced-control timer, then it is the internal clock source (CK_INT) that is selected as the clock. At this point CK_INT is CK_PSC.

### 12.2.2.2 External Clock Source Mode1

When the SMS domain is set to 111b, external clock source mode 1 is enabled. When external clock source 1 is enabled, TRGI is selected as the source of CK_PSC. it is worth noting that the source of TRGI also needs to be selected by configuring the TS domain. the TS domain can select the following types of pulses as clock sources.

1) Internal trigger (ITRx, x is 0,1,2,3).
2) Comparison of the signal after capturing channel 1 through the edge detector (TI1F_ED).
3) Comparison of signals TI1FP1, TI2FP2 of the capture channel.
4) The signal ETRF from the external clock pin input.

#### 12.2.2.3 External Clock Source Mode2

Use external trigger mode 2 to count on every rising or falling edge of the external clock pin input. When the ECE position is set, the external clock source mode 2 is used. when using the external clock source mode 2, ETRF is selected as CK_PSC. the ETR pin becomes ETRP after passing through the optional inverter (ETP), divider (ETPS), and then ETRF after passing through the filter (ETF).

With the ECE position bit and the SMS set to 111b, this is equivalent to the TS selecting ETRF as an input.

#### 12.2.2.4 Encoder Mode

Setting the SMS to 001b, 010b, 011b will enable the encoder mode. Enabling encoder mode allows you to select a specific level in TI1FP1 and TI2FP2 to signal the output with another jump edge as the signal. This mode is used when an external encoder is used. Refer to Section 12.3.10 for specific functions.

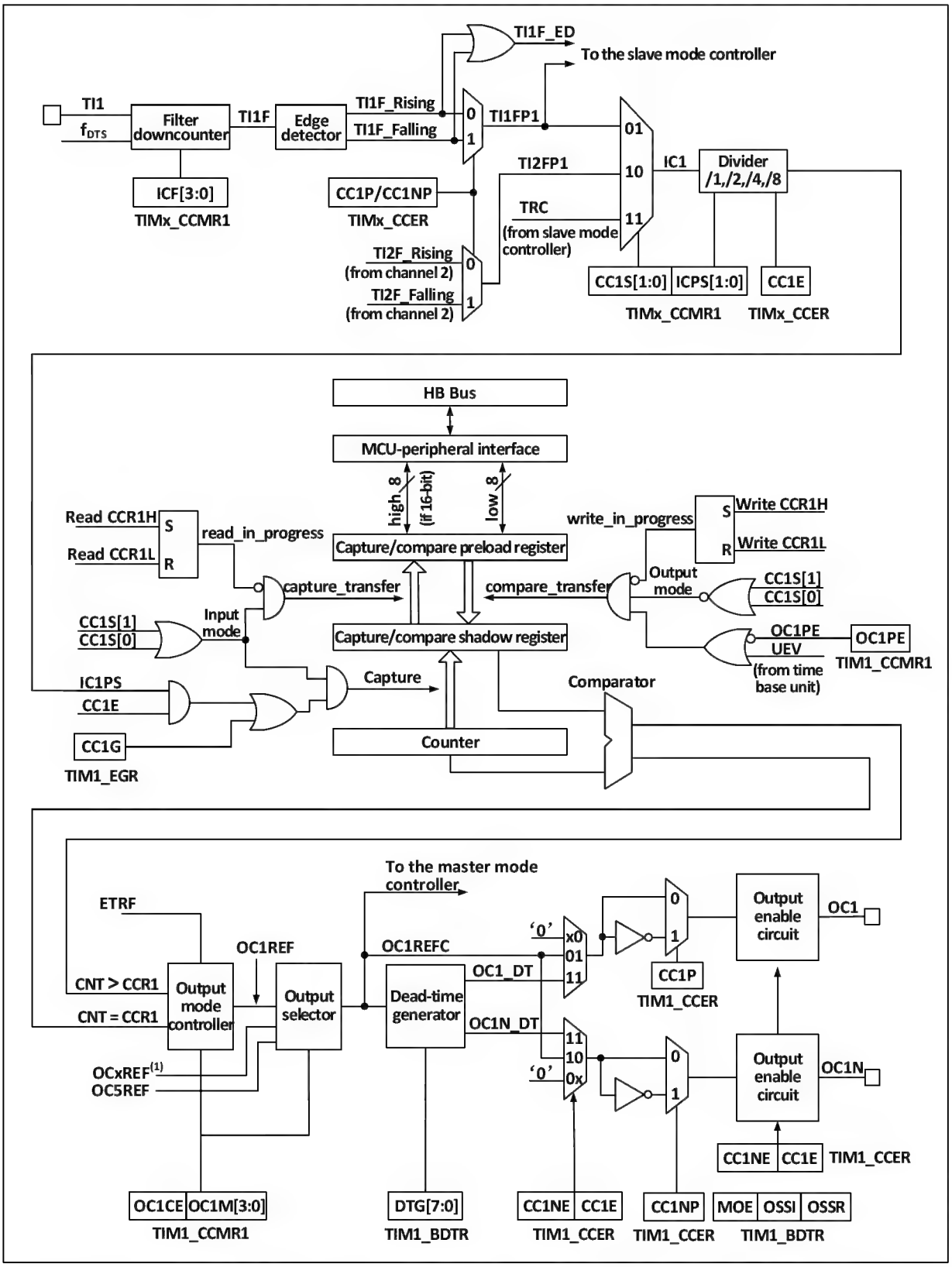
### 12.2.3 Counters and Peripherals

CK_PSC is input to the prescaler (PSC) for dividing. the PSC is 16-bit and the actual dividing factor is equal to the value of R16_TIMx_PSC + 1. CK_PSC goes through the PSC and becomes CK_INT. changing the value of R16_TIM1_PSC does not take effect in real time, but is updated to the PSC after an update event. the update event includes a UG bit clear and reset. The core of the timer is a 16-bit counter (CNT). CK_CNT is eventually fed to the CNT, which supports incremental count mode, decremental count mode, and incremental and decremental count modes, and has an Automatic Reload Register (ATRLR) that reloads the initial value for the CNT at the end of each count cycle. There is also an auxiliary counter that keeps track of the number of times the ATRLR reloads the initial value for the CNT and can generate a specific event when the number of times set in the Repeat Count Register (RPTCR) is reached.

### 12.2.4 Compare/Capture Channels and Peripherals

The core of the timer is the compare/capture register, which is complemented by digital filtering, frequency division and inter-channel multiplexing in the peripheral input section, comparator and output control in the output section.

Figure 12-3 Block diagram of the structure of the compare/capture channel




The structure block diagram of the compare/capture channel is shown in Figure 12-3. The signal is input from the channel x pin and optionally made as TIx (the source of TI1 can be more than just CH1, see the structure block diagram of timer 10-1), TI1 is passed through the filter (ICF[3:0]) to generate TI1F, and then divided into

TI1F_Rising and TI1F_Falling through the edge detector, these two signals are selected (CC1P) to generate TI1FP1, TI1FP1 and TI2FP1 from channel 2 are sent together to CC1S to select to become IC1, which is sent to the comparison capture register after ICPS dividing.

The compare capture register consists of a preload register and a shadow register, and the read/write process operates only on the preload register. In capture mode, the capture occurs on the shadow register and is then copied to the preload register; in compare mode, the contents of the preload register are copied to the shadow register, and then the contents of the shadow register are compared to the core counter (CNT).

## 12.3 Function and Implementation

The implementation of the complex functions of the advanced-control timer are all achieved by the operation of the timer's compare capture channel, clock input circuit and counter and peripheral parts. The clock input to the timer can come from multiple clock sources, including the input to the compare capture channel. The operation of the compare capture channel and clock source selection directly determines its function. The compare capture channel is bidirectional and can operate in both input and output modes.

### 12.3.1 Counter Mode

**Incremental counting modes**

In incremental counting mode, the counter counts from 0 to the automatic reload value (The contents of the R16_TIMx_ATRLR register) and then starts counting again from 0 and generates a counter overflow event.

If a repeat counter is used, an update event (UEV) will be generated when the number of repeats of the incremental count reaches the number programmed in the repeat counter register plus one (R16_TIMx_RPTCR+1). Otherwise, an update event will be generated each time the counter overflows.

An update event will also be generated when setting the TIMx_SWEVGR register to UG position 1 (Either by software or using the slave mode controller).

The UEV event can be disabled by software by setting the UDIS position 1 in the R16_TIMx_CTLR1 register. This prevents the shadow register from being updated when a new value is written to the preload register. No update event will be generated until the UDIS bit is written to 0. However, both the counter and the prescaler counter will start counting from 0 again (While the prescaler ratio remains unchanged). In addition, if the URS bit (Update Request Select) in the R16_TIMx_CTLR1 register is set to 1, setting UG location 1 will generate the update event UEV, but will not set the UIF flag to 1 (So no interrupts or DMA requests will be sent). This way, if the counter is cleared to zero when a capture event occurs, no update interrupt and no capture interrupt will be generated at the same time.

When an update event occurs, all registers will be updated and the update flag (UIF bit in the R16_TIMx_INTFR register) will be set to 1 (Depending on the URS bit):

1) the contents of the R16_TIMx_RPTCR register will be reloaded in the repeat counter
2) the auto-reload shadow register will be updated with the preload value (R16_TIMx_ATRLR)
3) Preload value will be reloaded in the prescaler buffer (Contents of R16_TIMx_PSC register)

**Decremental counting modes**

In decrement count mode, the counter decrements from the auto reload value (Contents of the R16_TIMx_ATRLR register) to a count of 0, then re-counts from the auto reload value and generates a counter underflow event.

If a repeat counter is used, an update event (UEV) will be generated when the number of repeats of the decrement count reaches the number programmed in the repeat counter register plus one (R16_TIMx_RPTCR+1). Otherwise,

an update event will be generated each time the counter overflows.
An update event will also be generated when setting the R16_TIMx_EGR register to UG position 1 (Either by software or using the slave mode controller).
The UEV update event can be disabled by software by setting the UDIS location 1 in the R16_TIMx_CTLR1 register. This prevents the shadow register from being updated when a new value is written to the preload register. No update event will be generated until a 0 is written to the UDIS bit. However, the counter will start counting again from the current auto-reload value and the prescaler counter will start counting again from 0 (But the prescaler ratio remains unchanged).
In addition, if the URS bit (Update Request Select) in the R16_TIMx_CTLR1 register has been set to 1, setting UG location 1 will generate the update event UEV, but will not set the UIF flag to 1 (So no interrupts or DMA requests will be sent). In this way, if the counter is cleared to zero when a capture event occurs, no update interrupt and no capture interrupt will be generated at the same time.
When an update event occurs, all registers will be updated and the update flag (UIF bit in the R16_TIMx_INTFR register) will be set to 1 (Depending on the URS bit):
1) The contents of the R16_TIMx_RPTCR register will be reloaded in the repeat counter
2) the preload value (Contents of the R16_TIMx_PSC register) will be reloaded in the buffer of the prescaler
3) The auto-reload active register will be updated with the preload value (Contents of the R16_TIMx_ATRLR register).
*Note: The Auto Reload register will be updated before the counter is reloaded, so that the next count cycle is the new cycle length we want.*

**Center alignment mode (Incremental/decremental counting)**
In center aligned mode, the counter counts from 0 to the auto-reload value (Contents of the R16_TIMx_ATRLR register) -1, generating a counter overflow event; it then counts down from the auto-reload value to 1 and generates a counter underflow event. After that the count starts again from 0.
The center-aligned mode is valid when the CMS bit in the R16_TIMx_CTLR1 register is not "00". When a channel is configured in output mode, its output compare interrupt flag will be set to 1 in the following modes: counter decrement count (Center-aligned mode 1, CMS="01"), counter increment count (Center-aligned mode 2, CMS="10 ") and Counter Incremental/Decremental Counting (Centre Alignment Mode 3, CMS="11").
In this mode, the DIR direction bit of the R16_TIMx_CTLR1 register is not writable, but is updated by hardware and indicates the current counter direction.
An update event is generated each time a counter overflow and underflow occurs, or an update event can also be generated by setting UG position 1 in the R16_TIMx_SWEVGR register (Either by software or using the slave mode controller). In this case the counter as well as the prescaler counter will start counting again from 0.
The UEV update event can be disabled by software by setting the UDIS position 1 in the R16_TIMx_CTLR1 register. This prevents the shadow register from being updated when a new value is written to the preload register. No update event will be generated until the UDIS bit is written to 0. However, the counter will still count incrementally and decrementally according to the current auto-reload value.
In addition, if the URS bit (Update Request Select) in the R16_TIMx_CTLR1 register has been set to 1, setting UG location 1 will generate a UEV update event, but will not set the UIF flag to 1 (And therefore no interrupts or DMA requests will be sent). This way, if the counter is cleared to zero when a capture event occurs, no update interrupt and no capture interrupt will be generated at the same time.
When an update event occurs, all registers will be updated and the update flag (UIF bit in the R16_TIMx_INTFR register) will be set to 1 (Depending on the URS bit):

1) the contents of the R16_TIMx_RPTCR register will be reloaded in the repeat counter
2) the preload value (Content of the R16_TIMx_PSC register) will be reloaded in the buffer of the prescaler
3) The auto-reload active register will be updated with the preload value (Contents of the R16_TIMx_ATRLR register). Note that if the update operation is triggered by a counter overflow, the auto-reload register is updated before the counter is reloaded, so that the next count cycle is the new cycle length we want (The counter is reloaded with the new value).

### 12.3.2 Input Capture Mode

The input capture mode is one of the basic functions of the timer. The principle of input capture mode is that a capture event occurs when a determined edge on the ICxPS signal is detected, and the current value of the counter is latched into the compare capture register (R16_TIMx_CHCTLRx). When a capture event occurs, CCxIF (in R16_TIMx_INTFR) is set, and if an interrupt or DMA is enabled, the corresponding interrupt or DMA is also generated. if CCxIF is already set when a capture event occurs, the CCxOF bit is set. CCxIF can be cleared by software, or by hardware by reading the compare capture register. CCxOF is cleared by software.

An example of channel 1 to illustrate the steps to use the input capture mode is as follows.

1) Configure the CCxS domain to select the source of the ICx signal. For example, set to 10b and select TI1FP1 as the source of IC1 instead of using the default setting, where the CCxS domain defaults to making the comparison capture module the output channel.
2) Configure the ICxF domain to set the digital filter for the TI signal. The digital filter will sample the signal at a determined frequency, a determined number of times, and then output a hop. This sampling frequency and number of times is determined by ICxF.
3) Configure the CCxP bit to set the polarity of the TIxFPx. For example, keeping the CC1P bit low and selecting rising edge jumps.
4) Configure the ICxPS domain to set the ICx signal to be the crossover factor between ICxPS. For example, keeping ICxPS at 00b, without crossover.
5) Configure the CCxE bit to allow capturing the value of the core counter (CNT) into the compare capture register. Set the CC1E bit.
6) Configure the CCxIE and CCxDE bits as needed to determine whether to allow enable interrupts or DMA.

This completes the comparison capture channel configuration.

When a captured pulse is input to TI1, the value of the core counter (CNT) is recorded in the compare capture register, CC1IF is set, and the CCIOF bit is set when CC1IF has been set before. If the CC1IE bit is set, then an interrupt is generated; if CC1DE is set, a DMA request is generated. An input capture event can be generated by software by writing the event generation register (TIMx_SWEVGR).

### 12.3.3 Compare Output Mode

The compare output mode is one of the basic functions of a timer. The principle of the compare output mode is to output a specific change or waveform when the value of the core counter (CNT) agrees with the value of the compare capture register. the OCxM field (In R16_TIMx_CHCTLRx) and the CCxP bit (in R16_TIMx_CCER) determine whether the output is a definite high or low level or a level flip. The CCxIF bit is also set when a compare coherent event is generated, an interrupt is generated if the CCxIE bit is pre-set, and a DMA request is generated if the CCxDE bit is pre-set.

The steps to configure to compare output mode are as follows:

1) Configure the clock source and auto-reload value of the core counter (CNT);
2) Set the count value to be compared to the compare capture register (R16_TIMx_CHxCVR);

3) setting the CCxIE bit if an interrupt needs to be generated;
4) disable the preload register of the compare register by keeping OCxPE to 0;
5) set the output mode, setting the OCxM field and the CCxP bit;
6) Enable the output, setting the CCxE bit;
7) Set the CEN bit to start the timer.

### 12.3.4 Forced Output Mode

The output pattern of the timer's compare capture channel can be forced by software to output a determined level without relying on comparison of the compare capture register's shadow register with the core counter.
This is done by setting OCxM to 100b, which forces OCxREF to low, or by setting OCxM to 101b, which forces OCxREF to high.
Note that by forcing OCxM to 100b or 101b, the comparison process of the internal core counters and compare capture registers is still going on, the corresponding flags are still set, and interrupts and DMA requests are still being generated.

### 12.3.5 PWM Input Mode

The PWM input mode is used to measure the duty cycle and frequency of PWM and is a special case of the input capture mode. The operation is the same as input capture mode except for the following differences: PWM occupies two compare capture channels and the input polarity of the two channels is set to opposite, one of the signals is set as trigger input and SMS is set to reset mode.
For example, to measure the period and frequency of the PWM wave input from TI1, the following operations are required.

1) Set TI1 (TI1FP1) to be the input of IC1 signal. Set CC1S to 01b.
2) Set TI1FP1 to rising edge active. Holding CC1P at 0.
3) Set TI1 (TI1FP2) as the input of IC2 signal. Set CC2S to 10b.
4) Select TI1FP2 to set to falling edge active. Set CC2P to 1.
5) Select TI1FP1 as the source of the clock source. set TS to 101b.
6) Set the SMS to reset mode, i.e. 100b.
7) Enables input capture. CC1E and CC2E are set.

Thus the value of compare capture register 1 is the period of the PWM, and the value of compare capture register 2 is its duty cycle.
*Note: Since only TI1FP1 and TI2FP2 are connected to the slave mode controller, only TIMx_CH1/TIMx_CH2 can be used in the PWM input mode.*

### 12.3.6 PWM Output Mode

PWM output mode is one of the basic functions of the timer. PWM output mode is most commonly used to determine the PWM frequency using the reload value and the duty cycle using the capture comparison register. Set 110b or 111b in the OCxM field to use PWM mode 1 or mode 2, set the OCxPE bit to enable the preload register, and finally set the ARPE bit to enable automatic reload of the preload register. Since the value of the preload register can only be sent to the shadow register when an update event occurs, the UG bit needs to be set to initialize all registers before the core counter starts counting. In PWM mode, the core counter and the compare capture register are always comparing, and depending on the CMS bit, the timer is able to output edge-aligned or center-aligned PWM signals.

- Edge aligned

When edge aligned is used, the core counter is incremented or decremented, and in the PWM mode 1 scenario, OCxREF is high when the core counter value is greater than the compare capture register, and low when the core counter value is less than the compare capture register (e.g., when the core counter grows to the value of R16_TIMx_ATRLR and reverts to all zeros).

- Center aligned

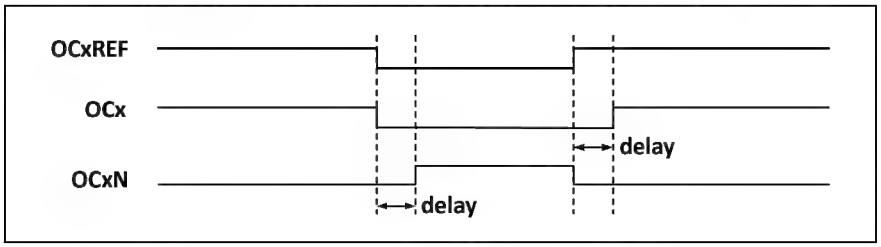
When using the center aligned modes, the core counter runs in alternating incremental and decremental count modes, and OCxREF makes rising and falling jumps when the values of the core counter and the compare capture register match. However, the comparison flags are set at different times in the three central alignment modes. When using the center aligned modes, it is best to generate a software update flag (Set the UG bit) before starting the core counter.

### 12.3.7 Complementary Outputs and Dead-time Insertion

The compare capture channel generally has two output pins (Compare capture channel 4 has only one output pin) and can output two complementary signals (OCx and OCxN). OCx and OCxN can be independently set for polarity via the CCxP and CCxNP bits, independently set for output enable via CCxE and CCxNE, and independently set for output enable via the MOE, OIS, OISN, OSSI, and OSSR bits for dead-time and other controls. Enabling the OCx and OCxN outputs simultaneously will insert a dead-time, and each channel has a 10-bit dead-time generator. OCx and OCxN are generated by the OCxREF association. If both OCx and OCxN are high active, then OCx is the same as OCxREF except that the rising edge of OCx is equivalent to OCxREF with a delay, and OCxN is the opposite of OCxREF in that its rising edge will have a delay relative to the falling edge of the reference signal. If the delay is greater than the effective output width, the corresponding pulse will not be generated.

The relationship between OCx and OCxN and OCxREF is illustrated in Figure 12-4, which shows the dead-time.

Figure 12-4 Complementary outputs and dead-time insertion




### 12.3.8 Brake Signal

When the brake signal is generated, the output enable signal and invalid level are modified according to the MOE, OIS, OISN, OSSI, and OSSR bits. However, OCx and OCxN will not be at the active level at any time. The source of the brake event can come from the brake input pin or it can be a clock failure event which is generated by the CSS (Clock Safety System).

After system reset, the brake function is disabled by default (MOE bit is low), and setting the BKE bit enables the brake function. The polarity of the input brake signal can be set by setting BKP, and the BKE and BKP signals can be written at the same time, and there is a delay of one HB clock before the actual writing, so you need to wait for one HB cycle to read the written value correctly.

At the presence of the selected level on the brake pin the system will generate the following actions.

1) The MOE bit is cleared asynchronously, setting the output to an invalid, idle or reset state, depending on the setting of the SOOI bit.
2) After the MOE has been cleared, each output channel outputs a level determined by OISx.

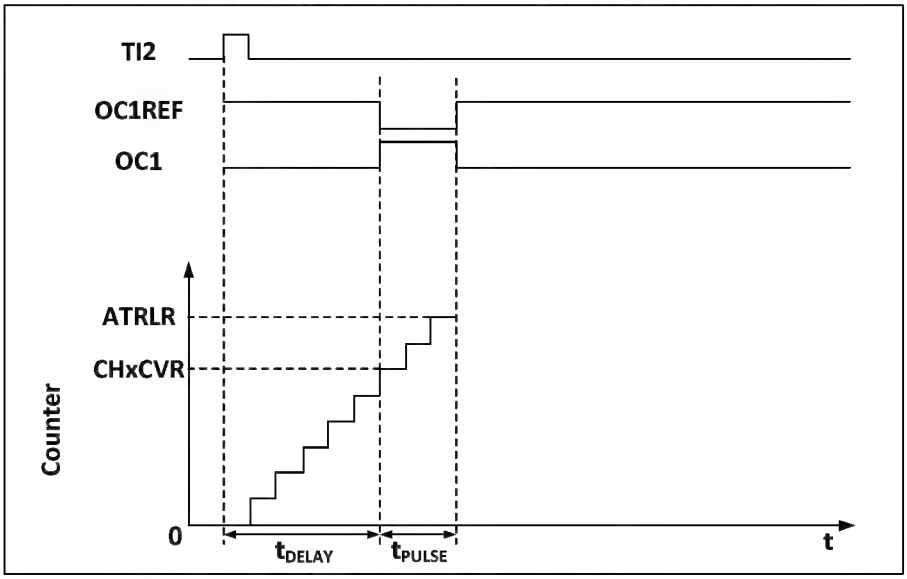
3) When using complementary outputs: the outputs are placed in a null state, depending on the polarity.
4) If the BIE is set, an interrupt is generated when the BIF is set; if the BDE bit is set, a DMA request is generated.
5) If the AOE is set, the MOE bit is automatically set at the next update event UEV.

### 12.3.9 One-pulse Mode

One-pulse mode (OPM) can be used to allow the microcontroller to respond to a specific event by causing it to generate a pulse after a delay, with the delay and width of the pulse programmable. Placing the OPM bit allows the core counter to stop when the next update event UEV is generated (Counter flips to 0).

As shown in Figure 10-5, a positive pulse of length Tpulse needs to be generated on OC1 after a delay Tdelay at the beginning of a rising edge detected on the TI2 input pin.

Figure 12-5 Generation of one pulse mode




1) Set TI2 to trigger. Setting the CC2S field to 01b to map TI2FP2 to TI2; setting the CC2P bit to 0b to set TI2FP2 as rising edge detection; setting the TS field to 110b to set TI2FP2 as trigger source; setting the SMS field to 110b to set TI2FP2 to be used to start the counter.
2) Tdelay is determined by the value of the Compare Capture Register, and Tpulse is determined by the value of the Auto Reload Value Register and the Compare Capture Register.

### 12.3.10 Encoder Mode

The encoder mode is a typical application of the timer and can be used to access the biphasic output of the encoder. The counting direction of the core counter is synchronized with the direction of the encoder's rotation axis, and each pulse output from the encoder will cause the core counter to add or subtract one. To use the encoder, set the SMS field to 001b (Count only on TI2 edge), 010b (Count only on TI1 edge) or 011b (Count on both TI1 and TI2 edges), connect the encoder to the input of comparison capture channels 1 and 2, and set a value for the reload value register, which can be set to a larger value. When in encoder mode, the internal compare capture register, prescaler, repeat count register, etc. of the timer are working normally. The following table shows the relationship between the counting direction and the encoder signal.

Table 12-1 Relationship between counting direction and encoder signal of timer encoder mode

| Counting effective edges | The level of relative signals | TI1FP1 signal edge | | TI2FP2 signal | |
|---|---|---|---|---|---|
| | | Rising edge | Falling edge | Rising edge | Falling edge |
| Counting at TI1 edge only | high | Downward counting | Upward counting | No count | |
| | low | Upward counting | Downward counting | | |
| Counting at TI2 edge only | high | No count | | Upward counting | Downward counting |
| | low | | | Downward counting | Upward counting |
| Double edge counting at TI1 and TI2 | high | Downward counting | Upward counting | Upward counting | Downward counting |
| | low | Upward counting | Downward counting | Downward counting | Upward counting |

### 12.3.11 Synchronization of TIMx Timers and External Triggers

The timer can be synchronized with an external trigger in reset mode, gated mode and trigger mode.

**Slave mode: reset mode**

The counter and its prescaler can be reinitialized in response to a trigger input event; if the URS bit of the R16_TIMx_CTLR1 register is low, an update event UEV is generated; all preload registers (R16_TIMx_ATRLR, R16_TIMx_CHxCVR) are then updated.

In the following example, the up counter is cleared to zero when a rising edge occurs on the TI1 input:

1) Configure channel 1 to detect the rising edge of TI1. Configure the input filter bandwidth (This example does not require any filter, so keep IC1F = 0000). There is no need to configure the capture divider as it is not used for trigger operation. the CC1S bit selects only the input capture source, i.e. CC1S=01 (In R16_TIMx_CCMR1). Write CC1P=0 and CC1NP='0' to the R16_TIMx_CCER register to verify polarity (Rising edge detection only).

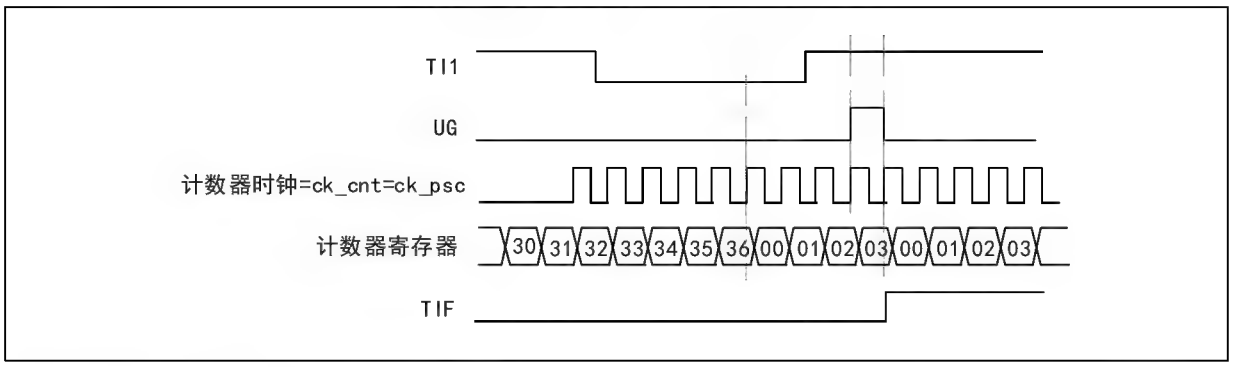
2) Write SMS=100 to R16_TIMx_SMCFGR to configure the timer to reset mode; write TS=101 to R16_TIMx_SMCFGR to select TI1 as input source.

3) Write CEN=1 to R16_TIMx_CTLR1 to start the counter.

The counter counts using the internal clock and then runs normally, when there is a TI1 rising edge the counter clears and starts counting again from 0. At the same time, the trigger flag TIF position 1, after enabling interrupt or DMA, allows an interrupt or DMA request to be sent. (Depends on the TIE (Interrupt Enable) bit and TDE (DMA Enable) bit in the R16_TIMx_DMAINTENR register).

The diagram below shows the action when the auto-reload register R16_TIMx_ARR = 0x36. The delay between the rising edge of TI1 and the actual counter reset is caused by the resynchronization circuitry at the TI1 input.

Figure 12-6 Control circuit in reset mode




**Slave mode: Gated mode**

The level of the input signal enables the counter. In the following example, the counter counts up only when TI1 is low:

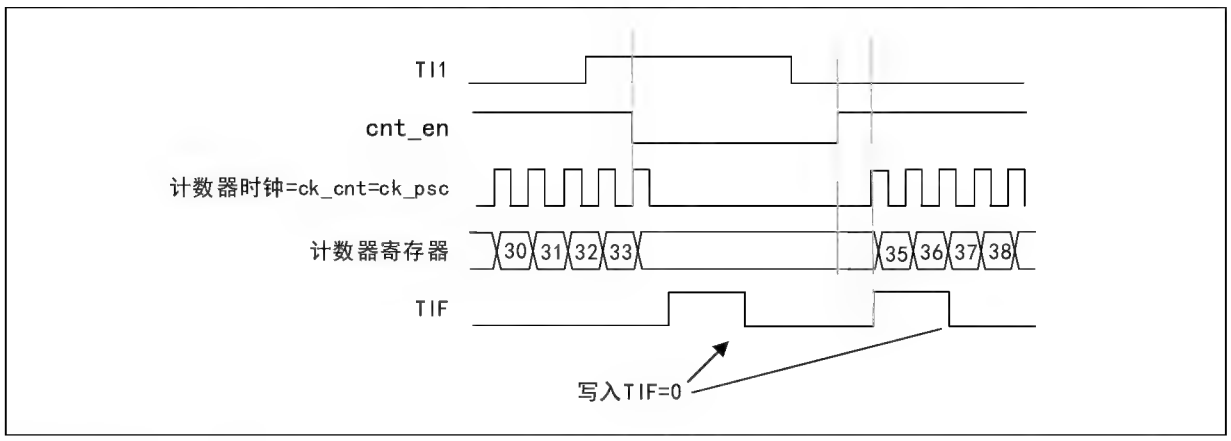
1) Configure channel 1 to detect a low level on TI1. Configure the input filter bandwidth (This example does not require any filter, so keep IC1F = 0000). There is no need to configure the capture divider as it is not used for trigger operation. the CC1S bit selects only the input capture source, i.e. CC1S=01 (in R16_TIMx_CCMR1). Write CC1P=1 and CC1NP='0' to the R16_TIMx_CCER register to verify polarity (detect low level only).

2) Write SMS=101 to R16_TIMx_SMCFGR to configure the timer for gated mode; write TS=101 to R16_TIMx_SMCFGR to select TI1 as input source.

3) Write CEN=1 to R16_TIMx_CTLR1 to start the counter. In gated mode, if CEN=0, the counter will not start regardless of the trigger input level.

As long as TI1 is low, the counter starts counting based on the internal clock and stops counting when TI1 goes high. When the counter starts or stops it sets TIF position 1 in R16_TIMx_INTFR. The delay between the rising edge of TI1 and the actual counter reset is caused by the resynchronization circuitry at the TI1 input.

Figure 12-7 Control circuit in gated mode




**Slave mode: Trigger mode**

An event on the selected input will enable the counter. In the following example, the up counter is activated when a rising edge occurs on the TI2 input:

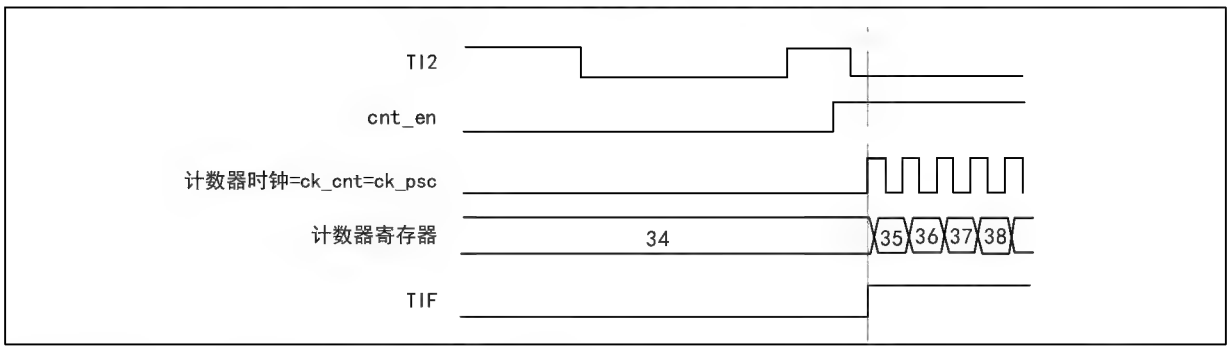
1) Configure channel 2 to detect the rising edge of TI2. Configure the input filter bandwidth (This example does not require any filter, so keep IC2F = 0000). There is no need to configure the capture divider as it is not used for trigger operation. the CC2S bit selects only the input capture source by setting CC2S=01 (In R16_TIMx_CCMR1). Write CC2P=1 and CC2NP='0' to the R16_TIMx_CCER register to verify polarity (Detect low level only)

2) Write SMS=110 to R16_TIMx_SMCFGR register to configure the timer to trigger mode; write TS=110 to

R16_TIMx_SMCFGR register to select TI2 as input source.

When there is a rising edge of TI2, the counter starts counting driven by the internal clock, while TIF is set to 1. The delay between the rising edge of TI2 and the actual counter start is caused by the resynchronization circuitry at the TI2 input.

Figure 12-8 Control circuit in trigger mode




**Slave mode: external clock mode 2 + trigger mode**

External clock mode 2 can be used in conjunction with another slave mode in addition to external clock mode 1 and encoder mode. In this case, the ETR signal is used as an input to the external clock and the other input can be used as a trigger input in reset mode, gated mode or trigger mode. It is not recommended to select ETR as TRGI via the TS bit of the R16_TIMx_SMCFGR register. in the following example, the up counter is incremented on each rising edge of ETR as soon as a rising edge occurs on TI1:

1) Configuring the R16_TIMx_SMCFGR register to configure the external trigger input circuit:

-ETF=0000: no filtering;

—ETPS=00: no prescaler used;

—ETP=0: detect the rising edge of ETR, set ECE=1 to enable external clock mode 2.

2) Configure channel 1 to detect the rising edge of TI:

—IC1F=0000: no filtering;

—no need to configure the capture divider as it is not used for trigger operation;

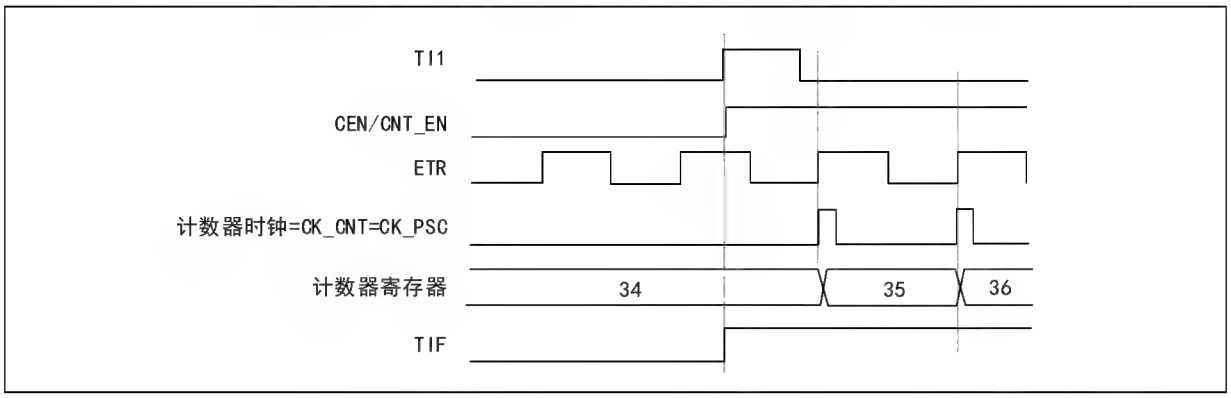
-setting CC1S=01 in the R16_TIMx_CHCTLR1 register to select the input capture source;

- set CC1P=0 in the R16_TIMx_CCER register to determine the polarity (Only rising edges are detected).

3) Write SMS=110 to R16_TIMx_SMCFGR register to configure the timer to trigger mode. Write TS=101 to the R16_TIMx_SMCFGR register to select TI1 as the input source.

When a rising edge occurs on TI1, the counter is enabled, TIF is set to 1 and the counter starts counting on the rising edge of ETR. the delay between the rising edge of the ETR signal and the actual counter reset is caused by the resynchronization circuit at the ETRP input.

Figure 12-9 Control circuit in external clock mode 2 + trigger mode

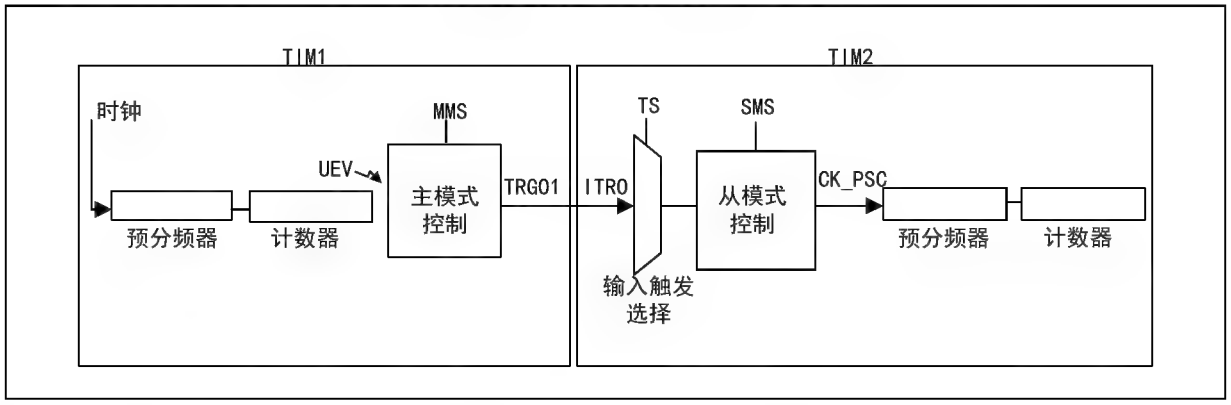



### 12.3.12 Timer Synchronization Mode

The TIMx timers are connected together from within to synchronize or cascade the timers. When a timer is configured in master mode, the counter of another timer configured in slave mode can be reset, started, stopped or clocked.

**Using one timer as a prescaler for another timer**

Figure 12-10 Master/slave timer example




For example, timer 1 can be configured as a prescaler for timer 2. To do this:

1) Configure Timer 1 in Master mode to output a periodic trigger signal every time an update event UEV occurs. If MMS=010 is written to R16_TIM1_CTLR2, TRGO1 will output a rising edge whenever an update event is generated.

2) To connect the TRGO1 output of Timer 1 to Timer 2, Timer 2 must be configured in slave mode, using ITR0 as the internal trigger. This can be selected via the TS bit in the R16_TIM2_SMCFGR register (Write TS=000).

3) The slave mode controller is then set to external clock mode 1 (Write SMS=111 in the R16_TIM2_SMCFGR register). In this way the clock for timer 2 will be supplied by the rising edge of the periodic trigger signal of timer 1 (Corresponding to the counter overflow of timer 1).

4) Finally both timers must be enabled simultaneously by setting the corresponding CEN bits (R16_TIMx_CTLR1 register) of both timers to 1.

*Note: If the OCx signal of timer 1 is selected as trigger output (MMS=1xx), the rising edge of this signal will be used to drive the counter of timer 2.*

**Using one timer to enable another timer**

In this example Timer 2 is enabled by comparing the output of Timer 1 with 1. Timer 2 counts according to the

divided internal clock only when OC1REF of Timer 1 is high. The clock frequency of both counters is based on CK_INT by prescaler performing a 3-way frequency (fCK_CNT=fCK_INT/3).

1) Configure timer 1 in main mode and send its output compare 1 reference signal (OC1REF) as trigger output (MMS=100 in R16_TIM1_CR2 register).

2) Configure Timer 1 for OC1REF waveform (R16_TIM1_CCMR1 register).

3) Configure Timer 2 to receive an input trigger from Timer 1 (TS=000 in R16_TIM2_SMCFGR register).

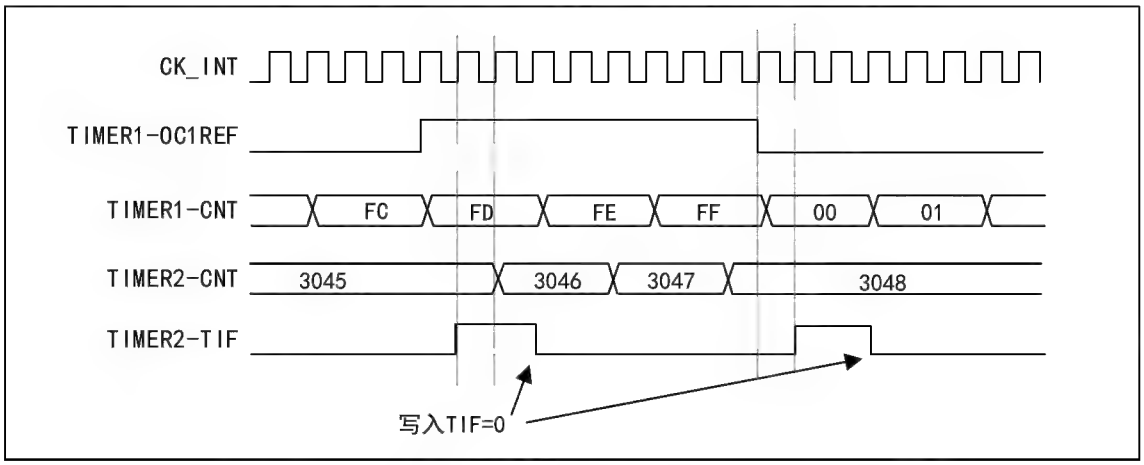
4) Configure Timer 2 to gated mode (SMS=101 in the R16_TIM2_SMCFGR register).

5) Enable Timer 2 by writing a "1" to the CEN bit (R16_TIM2_CTLR1 register).

6) Enable Timer 1 by writing "1" to the CEN bit (R16_TIM1_CTLR1 register).

*Note: The clock of Counter 2 is not synchronized with Counter 1, this mode only affects the counter enable signal of Timer 2.*

Figure 12-11 Gating Timer 2 using OC1REF of Timer 1




The counter and prescaler of timer 2 are not initialized before start-up. Therefore counting starts from the respective current value. Before starting Timer 1, both timers can be counted from the specified value by resetting them. This makes it possible to write any desired value into the timer counter. Both timers can be easily reset by software using the UG bit in the R16_TIMx_SWEVGR register.

In the next example, timer 1 is synchronized with timer 2. Timer 1 is in master mode and counts from 0. Timer 2 is in slave mode and counts from 0xE7. Both timers have the same prescale ratio. When timer 1 is disabled by writing "0" to the CEN bit in the R16_TIM1_CTLR1 register, timer 2 will stop:

1) Configure Timer 1 in main mode and send its output compare 1 reference signal (OC1REF) as the trigger output (MMS=100 in the R16_TIM1_CTLR2 register).

2) Configure Timer 1 for the OC1REF waveform (R16_TIM1_CHCTLR1 register).

3) Configure Timer 2 to receive an input trigger from Timer 1 (TS=000 in R16_TIM2_SMCFGR register).

4) Configure Timer 2 to gated mode (SMS=101 in the R16_TIM2_SMCFGR register).

5) Reset Timer 1 by writing "1" to the UG bit (R16_TIM1_SWEVGR register).

6) Reset Timer 2 by writing "1" to the UG bit (R16_TIM2_SWEVGR register).

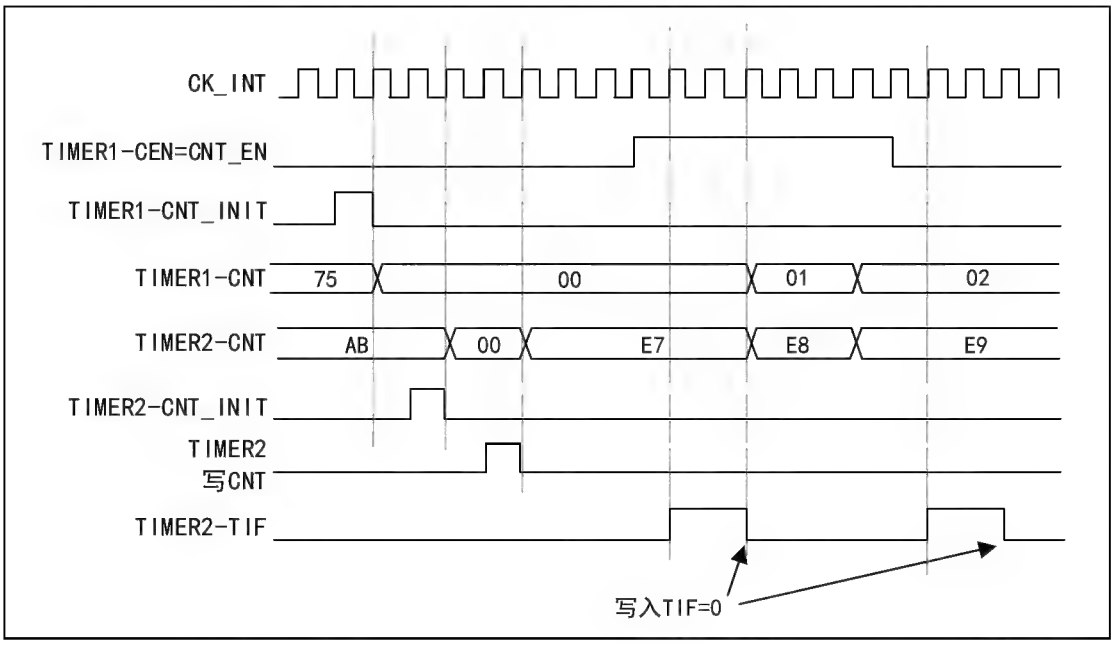
7) Initialize Timer 2 to 0xE7 by writing "0xE7" to Timer 2's counter (R16_TIM2_CNTL).

8) Enable Timer 2 by writing "1" to the CEN bit (R16_TIM2_CTLR1 register).

9) Enable Timer 1 by writing "1" to the CEN bit (R16_TIM1_CTLR1 register).

10) Stop Timer 1 by writing "0" to the CEN bit (R16_TIM1_CTLR1 register).

Figure 12-12 Gating Timer 2 using Timer 1's enable signal




**Using one timer to start another timer**

This example uses the update event of Timer 1 to enable Timer 2. As soon as Timer 1 generates an update event, Timer 2 starts counting from the current value (Which may not be 0) according to the internal clock after dividing the frequency. When Timer 2 receives a trigger signal, its CEN bit is automatically set to 1 and the counter starts counting until a "0" is written to the CEN bit of the R16_TIM2_CTLR1 register and the counter stops counting. The clock frequency of both counters is based on CK_INT and is divided by 3 through a prescaler (fCK_CNT=fCK_INT/3).

1) Configure timer 1 in main mode and send its update event (UEV) as trigger output (MMS=010 in R16_TIM1_CTLR2 register).

(MMS=010 in R16_TIM1_CTLR2 register).

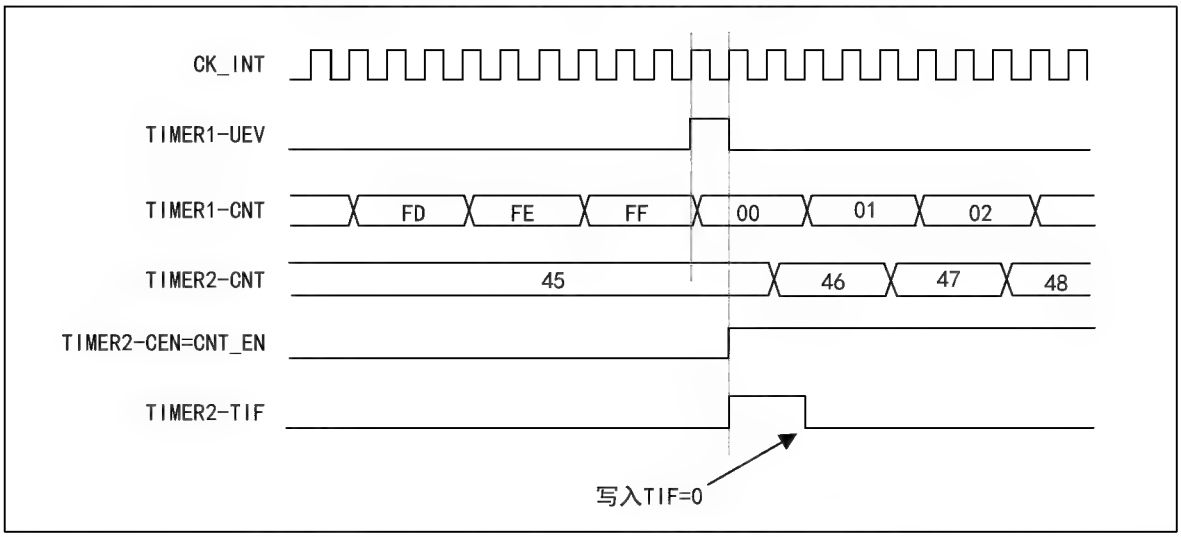
2) Configure the period of timer 1 (R16_TIM1_ATRLR register).

3) Configure Timer 2 to receive an input trigger from Timer 1 (TS=000 in the R16_TIM2_SMCFGR register).

4) Configure Timer 2 to trigger mode (SMS=110 in the R16_TIM2_SMCFGR register).

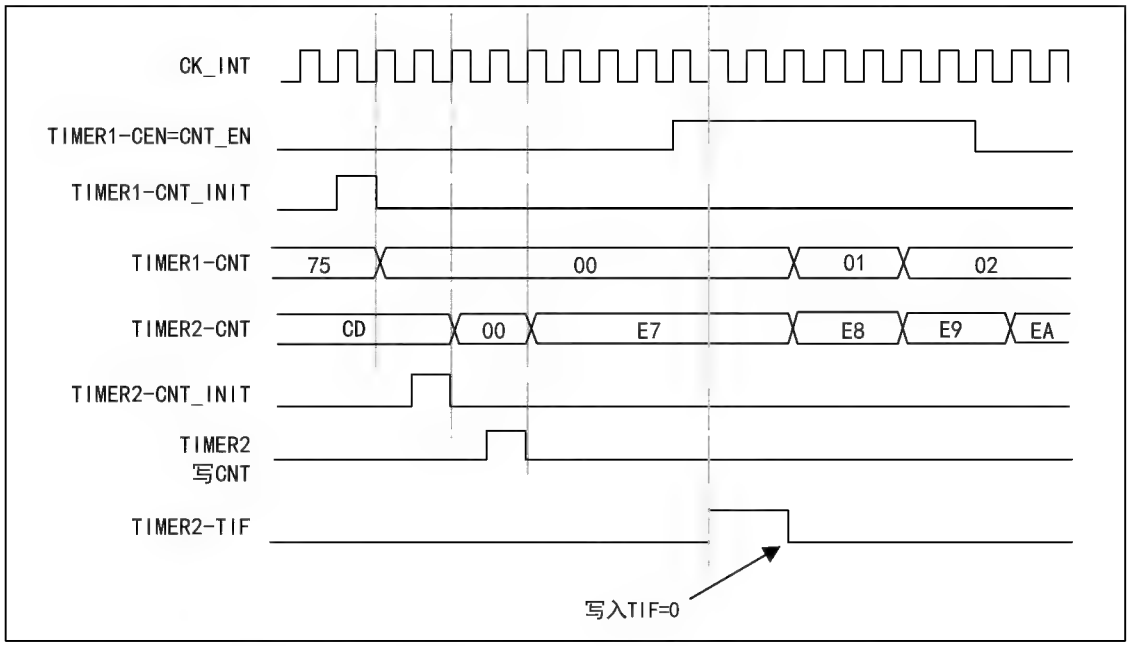
5) Start Timer 1 by writing "1" to the CEN bit (R16_TIM1_CTLR1 register).

Figure 12-13 Triggering Timer 2 using Timer 1 update event




As shown in the example above, the user can initialize both counters before starting to count. Figure 12-14 shows the counting behaviour with the same configuration as Figure 12-13, except in trigger mode (SMS=110 in the R16_TIM2_SMCFGR register) rather than gated mode.

Figure 12-14 Triggering Timer 2 using Timer 1's enable signal




**Using one timer as a prescaler for another timer**

For example, timer 1 can be configured as a prescaler for timer 2. To do this:

1) Configure Timer 1 in main mode, sending its update event (UEV) as a trigger output (MMS=010 in the R16_TIM1_CTLR2 register). This will then output a periodic signal each time the counter overflows.

2) Configure the period of timer 1 (R16_TIM1_ATRLR register).

3) Configure Timer 2 to receive an input trigger from Timer 1 (TS=000 in the R16_TIM2_SMCFGR register).

4) Configure Timer 2 for external clock mode (SMS=111 in the R16_TIM2_SMCFGR register).

5) Start Timer 2 by writing a "1" to the CEN bit (R16_TIM2_CTLR1 register).

6) Timer 1 is started by writing "1" to the CEN bit (R16_TIM1_CTLR1 register).

**Synchronized start of 2 timers using an external trigger**

In this example, Timer 1 is enabled when there is a rising edge on the TI1 input of Timer 1, and Timer 2 is enabled at the same time as Timer 1. To ensure that the two counters are aligned, Timer 1 must be configured in master/slave mode (Corresponding to TI1 as slave and Timer 2 as master):

1) Configure Timer 1 in Master mode, sending its enable signal as a trigger output (MMS=001 in the R16_TIM1_CTLR2 register).

2) Configure Timer 1 as Slave mode to receive the input trigger from TI1 (TS=100 in R16_TIM1_SMCFGR register).

3) Configure Timer 1 to trigger mode (SMS=110 in the R16_TIM1_SMCFGR register).

4) Configure Timer 1 to Master/Slave mode by writing MSM=1 (R16_TIMx_SMCR register).

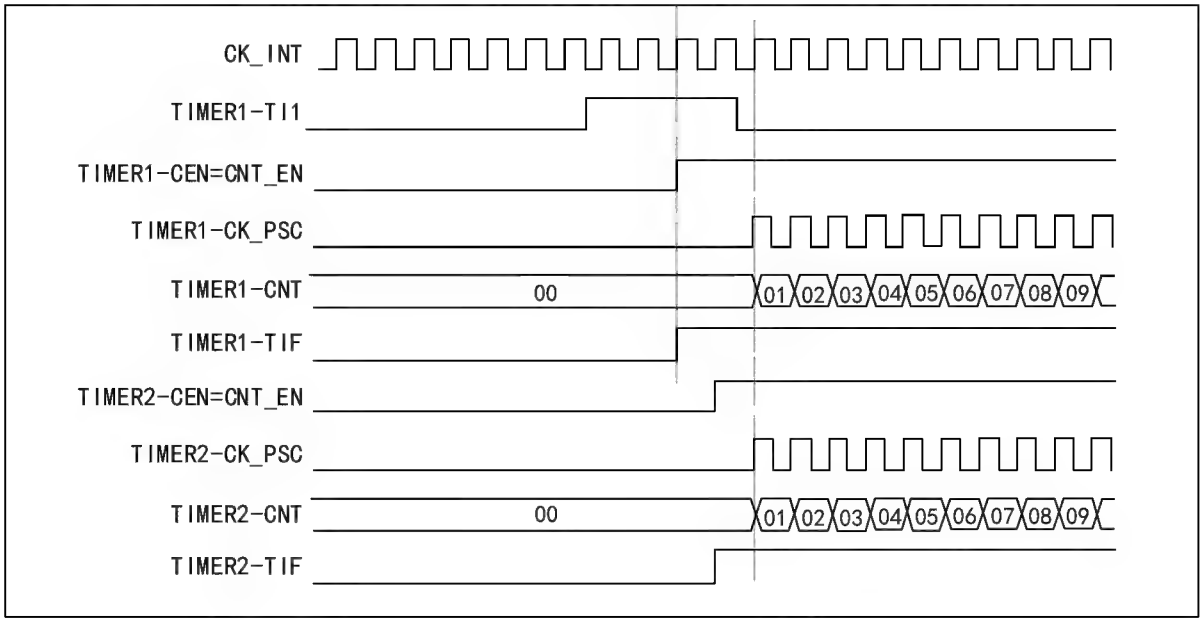
5) Configure Timer 2 to receive an input trigger from Timer 1 (TS=000 in R16_TIM2_SMCFGR register).

6) Configure Timer 2 to trigger mode (SMS=110 in the R16_TIM2_SMCFGR register).

When a rising edge occurs on TI1 (Timer 1), both counters start counting synchronously according to the internal clock and both TIF flags are set to 1.

*Note: In this example, both timers are initialized (By setting their respective UG positions to 1) prior to start-up. Both counters count from 0, but an offset can easily be inserted between the two by writing to either counter register (R16_TIMx_CNT). It may be noted that the master/slave mode creates a delay between CNT_EN and CK_PSC for timer 1.*

Figure 12-15 Triggering Timer 1 and Timer 2 using Timer 1's TI1 input




Timers are capable of outputting clock pulses (TRGO) and also receiving inputs from other timers (ITRx). The source of ITRx (TRGO from other timers) is different for different timers. The timer internal trigger connections are shown in Table 12-2.

Table 12-2 TIMx internal trigger connections

| Slave timer | ITR0(TS=000) | ITR1(TS=001) | ITR2(TS=010) | ITR3(TS=011) |
|---|---|---|---|---|
| TIM1 | 0 | TIM2_TRGO | 0 | 0 |
| TIM2 | TIM1_TRGO | 0 | 0 | 0 |

### 12.3.13 Debug Mode

When the system enters debug mode, the timer continues to run or stops according to the settings of the DBG module.

## 12.4 Register Description

Table 12-3 TIM1-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_TIM1_CTLR1 | 0x40012C00 | Control register 1 | 0x0000 |
| R16_TIM1_CTLR2 | 0x40012C04 | Control register 2 | 0x0000 |
| R16_TIM1_SMCFGR | 0x40012C08 | Slave mode control register | 0x0000 |
| R16_TIM1_DMAINTENR | 0x40012C0C | DMA/interrupt enable register | 0x0000 |
| R16_TIM1_INTFR | 0x40012C10 | Interrupt status register | 0x0000 |
| R16_TIM1_SWEVGR | 0x40012C14 | Event generation register | 0x0000 |
| R16_TIM1_CHCTLR1 | 0x40012C18 | Compare/capture control register 1 | 0x0000 |
| R16_TIM1_CHCTLR2 | 0x40012C1C | Compare/capture control register 2 | 0x0000 |
| R16_TIM1_CCER | 0x40012C20 | Compare/capture enable register | 0x0000 |
| R16_TIM1_CNT | 0x40012C24 | Counters | 0x0000 |
| R16_TIM1_PSC | 0x40012C28 | Counting clock prescaler | 0x0000 |
| R16_TIM1_ATRLR | 0x40012C2C | Auto-reload value register | 0xFFFF |
| R16_TIM1_RPTCR | 0x40012C30 | Repeat count Register | 0x0000 |
| R32_TIM1_CH1CVR | 0x40012C34 | Compare/capture register 1 | 0x00000000 |
| R32_TIM1_CH2CVR | 0x40012C38 | Compare/capture register 2 | 0x00000000 |
| R32_TIM1_CH3CVR | 0x40012C3C | Compare/capture register 3 | 0x00000000 |
| R32_TIM1_CH4CVR | 0x40012C40 | Compare/capture register 4 | 0x00000000 |
| R16_TIM1_BDTR | 0x40012C44 | Brake and deadtime registers | 0x0000 |
| R16_TIM1_DMACFGR | 0x40012C48 | DMA control register | 0x0000 |
| R16_TIM1_DMAADR | 0x40012C4C | DMA address register for continuous mode | 0x0000 |
| R16_TIM1_SPEC | 0x40012C50 | SPEC register | 0x0000 |

Table 12-4 TIM1-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_TIM2CTLR1 | 0x40000000 | Control register 1 | 0x0000 |
| R16_TIM2CTLR2 | 0x40000004 | Control register 2 | 0x0000 |
| R16_TIM2_SMCFGR | 0x40000008 | Slave mode control register | 0x0000 |
| R16_TIM2_DMAINTENR | 0x4000000C | DMA/interrupt enable register | 0x0000 |
| R16_TIM2_INTFR | 0x40000010 | Interrupt status register | 0x0000 |
| R16_TIM2_SWEVGR | 0x40000014 | Event generation register | 0x0000 |
| R16_TIM2_CHCTLR1 | 0x40000018 | Compare/capture control register 1 | 0x0000 |
| R16_TIM2_CHCTLR2 | 0x4000001C | Compare/capture control register 2 | 0x0000 |
| R16_TIM2_CCER | 0x40000020 | Compare/capture enable register | 0x0000 |
| R16_TIM2_CNT | 0x40000024 | Counters | 0x0000 |
| R16_TIM2_PSC | 0x40000028 | Counting clock prescaler | 0x0000 |
| R16_TIM2_ATRLR | 0x4000002C | Auto-reload value register | 0xFFFF |
| R16_TIM2_RPTCR | 0x40000030 | Repeat Count Register | 0x0000 |
| R32_TIM2_CH1CVR | 0x40000034 | Compare/capture register 1 | 0x00000000 |
| R32_TIM2_CH2CVR | 0x40000038 | Compare/capture register 2 | 0x00000000 |
| R32_TIM2_CH3CVR | 0x4000003C | Compare/capture register 3 | 0x00000000 |
| R32_TIM2_CH4CVR | 0x40000040 | Compare/capture register 4 | 0x00000000 |
| R16_TIM2BDTR | 0x40000044 | Brake and deadtime registers | 0x0000 |
| R16_TIM2DMACFGR | 0x40000048 | DMA control register | 0x0000 |

| | | | |
|---|---|---|---|
| R16_TIM2DMAADR | 0x4000004C | DMA address register for continuous mode | 0x0000 |
| R16_TIM2_SPEC | 0x40000050 | SPEC register | 0x0000 |

### 12.4.1 Control Register 1 (TIMx_CTLR1) (x=1/2)

Offset address: 0x00

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAPLVL | CAPOV | Reserved | CMP_BK | Reserved | | CKD[1:0] | | ARPE | CMS[1:0] | | DIR | OPM | URS | UDIS | CEN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | CAPLVL | RW | In double-edge capture mode, the capture level indication is enabled.<br>1: Enable the indication function;<br>0: Disable the indication function.<br>*Note: When enabled, CHxCVR[16] indicates the level corresponding to the capture value.* | 0 |
| 14 | CAPOV | RW | Capture value mode configuration.<br>1: The CHxCVR value is 0xFFFF when a counter overflow is generated before capture;<br>0: The capture value is the value of the actual counter. | 0 |
| 13 | Reserved | RO | Reserved. | 0 |
| 12 | CMP_BK | RW | Voltage comparator results are used directly for brake input enable (Supported by TIM1 only):<br>If CMP_BK is 1 and EN3 of OPA_CTLR2 is 1, the brake input signal comes from the CMPA3 output channel;<br>If CMP_BK is 1 and EN3 of OPA_CTLR2 is 0, the brake input signal comes from the CMPA1 output channel;<br>Otherwise, the brake input signal comes from the external corresponding I/O pins. | 0 |
| [11:10] | Reserved | RO | Reserved. | 0 |
| [9:8] | CKD[1:0] | RW | These 2 bits define the division ratio between the timer clock (CK_INT) frequency, the dead time and the sampling clock used by the dead time generator and the digital filter (ETR,TIx).<br>00: Tdts=Tck_int<br>01: Tdts = 2 × Tck_int<br>10: Tdts = 4 × Tck_int<br>11: Reserved. | 0 |
| 7 | ARPE | RW | Auto-reload preload enable bit.<br>1: Enable Auto-reload Value Register (ATRLR).<br>0: Disable Auto-reload Value Register (ATRLR). | 0 |
| [6:5] | CMS[1:0] | RW | Center aligned mode selection.<br>00: Edge-aligned mode. The counter counts up or down based on the direction bit (DIR).<br>01: Center-aligned mode 1. The counter counts up and down alternately. The output compare interrupt flag bit of the channel configured as output (CCxS=00 in the CHCTLRx register) is set only when the counter counts down.<br>10: Center-aligned mode 2. The counter counts up and down alternately. The output compare interrupt flag bit of the channel configured as output (CCxS=00 in the | 0 |

| | | | | |
|---|---|---|---|---|
| | | | CHCTLRx register) is set only when the counter counts up.<br>11: Center-aligned mode 3. The counter counts up and down alternately. The output compare interrupt flag bit of the channel configured as output (CCxS=00 in the CHCTLRx register) is set when the counter counts both up and down.<br>*Note: When the counter is enabled (CEN=1), the transition from edge-aligned mode to center-aligned mode is not allowed.* | |
| 4 | DIR | RW | Counting direction.<br>1: The counting mode of the counter is down counting;<br>0: The counting mode of the counter is up counting.<br>*Note: This bit is not valid when the counter is configured in center-aligned mode or encoder interface mode.* | 0 |
| 3 | OPM | RW | One-pulse mode.<br>1: The counter stops (Clear the CEN bit) when the next update event occurs.<br>0: The counter does not stop when the next update event occurs. | 0 |
| 2 | URS | RW | Update request source, by which the software selects the source of the UEV event.<br>1: If an update interrupt or DMA request is enabled, only an update interrupt or DMA request is generated if the counter overflows/underflows.<br>0: If an update interrupt or DMA request is enabled, an update interrupt or DMA request is generated by any of the following events.<br>-Counter overflow/underflow<br>-Setting the UG position<br>-Updates generated by the slave mode controller | 0 |
| 1 | UDIS | RW | Disable updates, the software allows/disables the generation of UEV events by means of this bit.<br>1: UEV is disabled. no update event is generated and the registers (ARR, PSC, CCRx) keep their values. If the UG bit is set or a hardware reset is issued from the mode controller, the counters and prescaler are reinitialized.<br>0: UEV is allowed. update (UEV) events are generated by any of the following events:<br>-Counter overflow/underflow<br>-Setting the UG position<br>- Updates generated by the slave mode controller<br>Registers with caches are loaded with their preloaded values. | 0 |
| 0 | CEN | RW | Enables the counter.<br>1: Enable the counter.<br>0: Disable the counter.<br>*Note: The external clock, gated mode and encoder mode will not work until the CEN bit is set in software. Trigger mode can automatically set the CEN bit in hardware.* | 0 |

### 12.4.2 Control Register 2 (TIMx_CTLR2) (x=1/2)

Offset address: 0x04

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | OIS4 | OIS3N | OIS3 | OIS2N | OIS2 | OIS1N | OIS1 | TI1S | MMS[2:0] | | | CCDS | CCUS | Reserved | CCPC |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | Reserved | RO | Reserved. | 0 |
| 14 | OIS4 | RW | Output idle state 4.<br>1: When MOE=0, if OC4N is implemented, OC1=1 after deadband;<br>0: When MOE=0, if OC4N is implemented, OC1=0 after deadband.<br>*Note: This bit cannot be modified after LOCK (TIMx_BDTR register) level 1, 2 or 3 has been set.* | 0 |
| 13 | OIS3N | RW | Output idle state 3.<br>1: OC1N = 1 after the dead zone when MOE = 0.<br>0: When MOE=0, OC1N=0 after dead zone.<br>*Note: This bit cannot be modified after the LOCK (TIMx_BDTR register) level 1, 2 or 3 has been set.* | 0 |
| 12 | OIS3 | RW | Output idle state 3, see OIS4. | 0 |
| 11 | OIS2N | RW | Output idle state 2, see OIS3N. | 0 |
| 10 | OIS2 | RW | Output idle state 2, see OIS4. | 0 |
| 9 | OIS1N | RW | Output idle state 1, see OIS3N. | 0 |
| 8 | OIS1 | RW | Output idle state 1, see OIS4. | 0 |
| 7 | TI1S | RW | TI1 selection.<br>1: TIMx_CH1, TIMx_CH2 and TIMx_CH3 pins connected to TI1 input (XOR combination).<br>0: TIMx_CH1 pin is connected directly to TI1 input. | 0 |
| [6:4] | MMS[2:0] | RW | Master mode selection: These 3 bits are used to select the synchronization information (TRGO) sent to the slave timer in master mode. The possible combinations are as follows.<br>000: The UG bit of the Reset-TIMx_EGR register is used as the trigger output (TRGO). In the case of a reset generated by a trigger input (From a mode controller in reset mode), there is a delay in the signal on TRGO relative to the actual reset.<br>001: Enable - The counter enable signal CNT_EN is used as a trigger output (TRGO). Sometimes it is necessary to start multiple timers at the same time or to control the enable from timers over a period of time. The counter enable signal is generated by the logical or of the trigger input signal in CEN control bit and gated mode. When the counter enable signal is controlled by a trigger input, there is a delay on TRGO unless master/slave mode is selected (See the description of the MSM bit in the TIMx_SMCR register).<br>010: Update - The update event is selected as a trigger input (TRGO). For example, the clock of a master timer may be used as a prescaler for a slave timer.<br>011: comparison pulse - on the occurrence of a capture or a successful comparison, when the CC1IF flag is to be set (Even if it is already high), the trigger output | 0 |

| | | | | |
|---|---|---|---|---|
| | | | sends a positive pulse (TRGO).<br>100: The comparison-OC1REF signal is used as a trigger output (TRGO).<br>101: The comparison-OC2REF signal is used as a trigger output (TRGO).<br>110: The comparison-OC3REF signal is used as a trigger output (TRGO).<br>111: The compare-OC4REF signal is used as the trigger output (TRGO). | |
| 3 | CCDS | RW | Capture compare DMA selection.<br>1: Send a DMA request for CHxCVR when an update event occurs.<br>0: Generate a DMA request for CHxCVR when CHxCVR occurs. | 0 |
| 2 | CCUS | RW | Compare capture control update selection bits.<br>1: If CCPC is set, they can be updated by setting the COM bit or a rising edge on TRGI.<br>0: If the CCPC is set, they can only be updated by setting the COM bit.<br>*Note: This bit only works for channels with complementary outputs.* | 0 |
| 1 | Reserved | RO | Reserved. | 0 |
| 0 | CCPC | RW | Compare capture preload control bits.<br>1: The CCxE, CCxNE and OCxM bits are preloaded and when this bit is set they are only updated when the COM bit is set.<br>0: CCxE, CCxNE and OCxM bits are not preloaded.<br>*Note: This bit only works for channels with complementary outputs.* | t |

### 12.4.3 Slave Mode Control Register (TIMx_SMCFGR) (x=1/2)

Offset address: 0x08

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETP | ECE | ETPS[1:0] | | ETF[3:0] | | | | MSM | TS[2:0] | | | Reserved | SMS[2:0] | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | ETP | RO | ETR trigger polarity selection, this bit selects whether to input ETR directly or to input the inverse of ETR.<br>1: Invert ETR, low or falling edge active;<br>0: ETR, active high or rising edge. | 0 |
| 14 | ECE | RW | External clock mode 2 enable selection.<br>1: Enables external clock mode 2.<br>0: Disable external clock mode 2.<br>*Note 1: Slave mode can be used simultaneously with external clock mode 2: reset mode, gated mode and trigger mode; however, TRGI cannot be connected to ETRF in this case (TS bit cannot be '111').*<br>*Note 2: When both external clock mode 1 and external clock mode 2 are enabled, the external clock input is ETRF.* | 0 |
| [13:12] | ETPS[1:0] | RW | The external trigger signal (ETRP) divides the frequency of this signal, which cannot exceed a maximum of 1/4 of the TIMxCLK frequency, and can be downconverted through this domain. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 00: Prescaler off.<br>01: ETRP frequency divided by 2.<br>10: ETRP frequency divided by 4.<br>11: ETRP frequency divided by 8. | |
| [11:8] | ETF[3:0] | RW | Externally triggered filtering, in fact, the digital filter is an event counter, which uses a certain sampling frequency to record up to N events and then produces a jump in the output.<br>0001: Sampling frequency Fsampling=Fck_int, N=2.<br>0010: Sampling frequency Fsampling=Fck_int, N=4.<br>0011: Sampling frequency Fsampling=Fck_int, N=8.<br>0100: Sampling frequency Fsampling = Fdts/2, N = 6.<br>0101: Sampling frequency Fsampling = Fdts/2, N = 8.<br>0110: Sampling frequency Fsampling = Fdts/4, N = 6.<br>0111: Sampling frequency Fsampling = Fdts/4, N = 8.<br>1000: Sampling frequency Fsampling = Fdts/8, N = 6.<br>1001: Sampling frequency Fsampling = Fdts/8, N = 8.<br>1010: Sampling frequency Fsampling = Fdts/16, N = 5.<br>1011: Sampling frequency Fsampling = Fdts/16, N = 6.<br>1100: Sampling frequency Fsampling = Fdts/16, N = 8.<br>1101: Sampling frequency Fsampling = Fdts/32, N = 5.<br>1110: Sampling frequency Fsampling = Fdts/32, N = 6.<br>1111: Sampling frequency Fsampling=Fdts/32, N=8. | 0 |
| 7 | MSM | RW | Master/slave mode selection.<br>1: The event on the trigger input (TRGI) is delayed to allow perfect synchronization between the current timer (via TRGO) and its slave timer. This is useful when the synchronization of several timers to a single external event is required.<br>0: Does not function. | 0 |
| [6:4] | TS[2:0] | RW | Trigger selection field, these 3 bits select the trigger input source used to synchronize the counter.<br>000: Internal trigger 0 (ITR0).<br>001: Internal trigger 1 (ITR1).<br>010: Internal trigger 2 (ITR2).<br>011: Internal trigger 3 (ITR3).<br>100: Edge detector of TI1 (TI1F_ED).<br>101: Filtered timer input 1 (TI1FP1).<br>110: Filtered timer input 2 (TI2FP2).<br>111: External trigger input (ETRF).<br>The above only changes when SMS is 0.<br>*Note: See Table 12-2 for details.* | 0 |
| 3 | Reserved | RO | Reserved. | 0 |
| [2:0] | SMS[2:0] | RW | Input mode selection field. Selects the clock and trigger mode of the core counter.<br>000: Driven by the internal clock CK_INT.<br>001: Encoder mode 1, where the core counter increments or decrements the count at the edge of TI2FP2 depending on the level of TI1FP1.<br>010: Encoder mode 2, where the core counter increments or decrements the count at the edge of TI1FP1, depending on the level of TI2FP2.<br>011: Encoder mode 3, where the core counter increments and decrements the count on the edges of TI1FP1 and TI2FP2 depending on the input level of another signal;<br>100: reset mode, where the rising edge of the trigger input (TRGI) will initialize the counter and generate a signal to | 0 |

| | | | | |
|---|---|---|---|---|
| | | | update the registers.<br>101: Gated mode, when the trigger input (TRGI) is high, the counter clock is turned on; at the trigger input becomes low, the counter is stopped, and the counter starts and stops are controlled.<br>110: Trigger mode, where the counter is started on the rising edge of the trigger input TRGI and only the start of the counter is controlled.<br>111: External clock mode 1, rising edge of the selected trigger input (TRGI) drives the counter. | |

### 12.4.4 DMA/Interrupt Enable Register (TIMx_DMAINTENR) (x=1/2)

Offset address: 0x0C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | TDE | COMDE | CC4DE | CC3DE | CC2DE | CC1DE | UDE | BIE | TIE | COMIE | CC4IE | CC3IE | CC2IE | CC1IE | UIE |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | Reserved | RO | Reserved | 0 |
| 14 | TDE | RW | Trigger DMA request enable bit.<br>1: Enable trigger DMA request;<br>0: Disable trigger DMA request. | 0 |
| 13 | COMDE | RW | COM DMA request enable bit.<br>1: Allow COM DMA requests;<br>0: Disable COM DMA requests. | 0 |
| 12 | CC4DE | RW | Compare/capture channel 4 DMA request enable bit.<br>1: Enable the DMA request of compare capture channel 4;<br>0: Disable the DMA request of compare capture channel 4. | 0 |
| 11 | CC3DE | RW | Compare/capture channel 3 DMA request enable bit.<br>1: Enable the DMA request of compare capture channel 3;<br>0: Disable the DMA request of compare capture channel 3. | 0 |
| 10 | CC2DE | RW | Compare/capture channel 2 DMA request enable bit.<br>1: Enable the DMA request of compare capture channel 2;<br>0: Disable the DMA request of compare capture channel 2. | 0 |
| 9 | CC1DE | RW | Compare/capture channel 1 DMA request enable bit.<br>1: Enable the DMA request of compare capture channel 1;<br>0: Disable the DMA request of compare capture channel 1. | 0 |
| 8 | UDE | RW | Update DMA request enable bit.<br>1: Enable the updated DMA request<br>0: Disable the updated DMA request. | 0 |
| 7 | BIE | RW | Brake interrupt enable bit.<br>1: Enable the brake interrupt;<br>0: Disable the brake interrupt. | 0 |
| 6 | TIE | RW | Trigger interrupt enable bit<br>1: Enable the trigger interrupt;<br>0: Disable the trigger interrupt. | 0 |

| | | | | |
|---|---|---|---|---|
| 5 | COMIE | RW | COM interrupt enable bit<br>1: Enable COM interrupt;<br>0: Disable COM interrupt. | 0 |
| 4 | CC4IE | RW | Compare capture channel 4 interrupt enable bit<br>1: Enable compare capture channel 4 interrupt;<br>0: Disable compare capture channel 4 interrupt. | 0 |
| 3 | CC3IE | RW | Compare capture channel 3 interrupt enable bit<br>1: Enable compare capture channel 3 interrupt;<br>0: Disable compare capture channel 3 interrupt. | 0 |
| 2 | CC2IE | RW | Compare capture channel 2 interrupt enable bit<br>1: Enable compare capture channel 2 interrupt;<br>0: Disable compare capture channel 2 interrupt. | 0 |
| 1 | CC1IE | RW | Compare capture channel 1 interrupt enable bit<br>1: Enable compare capture channel 1 interrupt;<br>0: Disable compare capture channel 1 interrupt. | 0 |
| 0 | UIE | RW | Update interrupt enable bit<br>1: Enable updated interrupt;<br>0: Disable updated interrupt. | 0 |

### 12.4.5 Interrupt Status Register (TIMx_INTFR) (x=1/2)

Offset address: 0x10

| 15 14 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | CC4OF | CC3OF | CC2OF | CC1OF | Reserved | BIF | TIF | COMIF | CC4IF | CC3IF | CC2IF | CC1IF | UIF |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:13] | Reserved | RO | Reserved | 0 |
| 12 | CC4OF | RW0 | Compare capture channel 4 over-capture flag bit. | 0 |
| 11 | CC3OF | RW0 | Compare capture channel 3 over-capture flag bit. | 0 |
| 10 | CC2OF | RW0 | Compare capture channel 2 over-capture flag bit. | 0 |
| 9 | CC1OF | RW0 | Compare capture channel 1 over-capture flag, only used when the compare capture channel is configured in input capture mode. This flag is set by hardware and cleared by software by writing 0.<br>1: When the counter value is captured into the capture compare register, the status of CC1IF has been set;<br>0: No over-capture occurs. | 0 |
| 8 | Reserved | RO | Reserved | 0 |
| 7 | BIF | RW0 | The brake interrupt flag bit, once the brake input is valid, by hardware for this position bit, can be cleared by software.<br>1: A set valid level is detected on the brake pin input.<br>0: No braking event is generated. | 0 |
| 6 | TIF | RW0 | Trigger interrupt flag bit, when a trigger event occurs by hardware to this location bit, by software to clear. Trigger events include the detection of a valid edge at the TRGI input from a mode other than gated, or any edge in gated mode.<br>1: Trigger event generation.<br>0: No trigger event is generated. | 0 |
| 5 | COMIF | RW0 | COM interrupt flag bit, this bit is set by hardware and cleared by software once a COM event is generated. com events including CCxE, CCxNE, OCxM are updated.<br>1: COM event generation. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: No COM event is generated. | |
| 4 | CC4IF | RW0 | Compare capture channel 4 interrupt flag bit. | 0 |
| 3 | CC3IF | RW0 | Compare capture channel 3 interrupt flag bit. | 0 |
| 2 | CC2IF | RW0 | Compare capture channel 2 interrupt flag bit. | 0 |
| 1 | CC1IF | RW0 | Compare capture channel 1 interrupt flag bit.<br>If the compare capture channel is configured in output mode.<br>This bit is set by hardware when the counter value matches the comparison value, except in centrosymmetric mode. This bit is cleared by software.<br>1: The value of the core counter matches the value of compare capture register 1;<br>0: No match occurs.<br>If compare capture channel 1 is configured as input mode. This bit is set by hardware when a capture event occurs, and it is cleared by software or by reading the compare capture register.<br>1: The counter value has been captured compare capture register 1.<br>0: No input capture is generated. | 0 |
| 0 | UIF | RW0 | Update interrupt flag bit, this bit is set by hardware when an update event is generated and cleared by software.<br>1: Update interrupt generation.<br>0: No update event is generated.<br>The following scenarios generate update events.<br>If UDIS = 0, when the repeat counter value overflows or underflows.<br>If URS = 0, UDIS = 0, when the UG bit is set, or when the counter core counter is reinitialized by software.<br>If URS = 0, UDIS = 0, when the counter CNT is reinitialized by a trigger event. | 0 |

### 12.4.6 Event Generation Register (TIMx_SWEVGR) (x=1/2)

Offset address: 0x14

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | BG | TG | COMG | CC4G | CC3G | CC2G | CC1G | UG |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | BG | WO | The brake event generation bit, which is set and cleared by software, is used to generate a brake event.<br>1: Generate a brake event. At this point, MOE=0, BIF=1, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA are generated.<br>0: No action. | 0 |
| 6 | TG | WO | The trigger event generation bit, which is set by software and cleared by hardware, is used to generate a trigger event.<br>1: Generate a trigger event, TIF is set, and the corresponding interrupts and DMAs are generated if enabled.<br>0: No action. | 0 |

| | | | | |
|---|---|---|---|---|
| 5 | COMG | WO | Compare capture control update generation bit. Generates a compare capture control update event. This bit is set by software and automatically cleared by hardware.<br>1: When CCPC = 1, allow updating of CCxE, CCxNE, OCxM bits.<br>0: No action.<br>*Note: This bit is only valid for channels with complementary outputs (Channels 1, 2, 3).* | 0 |
| 4 | CC4G | WO | Compare capture event generation bit 4. generates compare capture event 4. | 0 |
| 3 | CC3G | WO | Compare capture event generation bit 3. generates compare capture event 3. | 0 |
| 2 | CC2G | WO | Compare capture event generation bit 2. generates compare capture event 2. | 0 |
| 1 | CC1G | WO | Compare capture event generation bit 1. generates compare capture event 1.<br>This bit is set by software and cleared by hardware. It is used to generate a compare capture event.<br>1: Generate a compare capture event on compare capture channel 1.<br>If compare capture channel 1 is configured as output.<br>Set the CC1IF bit. Generate the corresponding interrupts and DMAs if they are enabled.<br>If compare capture channel 1 is configured as input.<br>The current core counter value is captured to compare capture register 1; set the CC1IF bit to generate the corresponding interrupts and DMAs if they are enabled; if CC1IF is already set, set the CC1OF bit.<br>0: No action. | 0 |
| 0 | UG | WO | Update event generation bit to generate an update event. This bit is set by software and is automatically cleared by hardware.<br>1: Initialize the counter and generate an update event.<br>0: No action.<br>*Note: The prescaler counter is also cleared to zero, but the prescaler factor remains unchanged. The core counter is cleared if in centrosymmetric mode or incremental counting mode; if in decremental counting mode, the core counter takes the value of the reload value register.* | 0 |

### 12.4.7 Compare/Capture Control Register 1 (TIMx_CHCTLR1) (x=1/2)

Offset address: 0x18

The channel can be used in input (Capture mode) or output (Compare mode), and the direction of the channel is defined by the corresponding CCxS bit. The other bits of this register have different roles in input and output modes. OCxx describes the function of the channel in output mode and ICxx describes the function of the channel in input mode.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OC2CE | OC2M[2:0] | | | OC2PE | OC2FE | CC2S[1:0] | | OC1CE | OC1M[2:0] | | | OC1PE | OC1FE | CC1S[1:0] | |
| IC2F[3:0] | | | | IC2PSC[1:0] | | | | IC1F[3:0] | | | | IC1PSC[1:0] | | | |

Compare mode (pin direction is output).

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | OC2CE | RW | Compare capture channel 2 clear enable bit.<br>1: Clear OC2REF bit zero once ETRF input is detected high;<br>0: OC2REF is not affected by ETRF input. | 0 |
| [14:12] | OC2M[2:0] | RW | Compare Capture Channel 2 mode setting field.<br>The 3 bits define the action of the output reference signal OC2REF, which determines the values of OC2, OC2N. OC2REF is active high, while the active levels of OC2 and OC2N depend on the CC2P, CC2NP bits.<br>000: Freeze. Comparison of the value of the capture register with the value of the comparison between the core counters does not work for OC1REF.<br>001: Force to set to valid level. Forcing OC1REF high when the core counter has the same value as the comparison capture register 2.<br>010: Force to set to invalid level. Forcing OC1REF low when the value of the core counter is the same as the comparison capture register 2.<br>011: Flip. Flips the level of OC1REF when the core counter is the same as the value of compare capture register 2.<br>100: Forced to invalid level. Forces OC1REF to low.<br>101: Forced to valid level. Force OC1REF to high.<br>110: PWM mode 1: When counting up, once the core counter is less than the value of the comparison capture register, channel 2 is an effective level, otherwise it is an invalid level; When counting down, once the core counter is greater than the value of the comparison capture register, channel 2 is at an invalid level (OC2REF=0), otherwise it is at an effective level (Oc2Ref = 1);<br>111: PWM mode 2: When counting up, once the core counter is less than the value of the comparison capture register, channel 2 is at an invalid level, otherwise it is at an effective level; When counting down, once the core counter is greater than the value of the comparison capture register, Channel 2 is active (OC2REF=1), otherwise it is inactive (OC2REF=0).<br>*Note: This bit cannot be modified once the LOCK level is set to 3 and CC1S=00b. In PWM mode 1 or PWM mode 2, the OC1REF level is changed only when the comparison result is changed or when switching from freeze mode to PWM mode in the output comparison mode.* | 0 |
| 11 | OC2PE | RW | Compare Capture Register 2 preload enable bit.<br>1: Enable the preload function of compare capture register 2, read and write operations only operate on the preload registers, the preload value of compare capture register 2 is loaded into the current shadow register when the update event comes;<br>0: Disable the preload function of compare capture register 2, compare capture register 2 can be written at any time, and the newly written value takes effect immediately.<br>*Note: Once the LOCK level is set to 3 and CC2S=00, this bit cannot be modified; PWM mode can be used* | 0 |

| | | | | |
|---|---|---|---|---|
| | | | *only in single pulse mode (OPM=1) without confirming the pre-load register, otherwise its action is not determined.* | |
| 10 | OC2FE | RW | Compare Capture Channel 2 fast enable bit, this bit is used to speed up the response of the compare capture channel output to a trigger input event.<br>1: The active edge of the input to the flipflop acts as if a comparison match has occurred. Therefore, the OC is set to the comparison level independent of the comparison result. The delay between the valid edge of the sample trigger and the output of the compare capture channel 2 is reduced to 3 clock cycles.<br>0: Based on the value of the counter and compare capture register 1, compare capture channel 2 operates normally, even if the flip-flop is open. The minimum delay to activate the compare capture channel 2 output is 5 clock cycles when the input of the flipflop has a valid edge.<br>OC2FE only works when the channel is configured to PWM1 or PWM2 mode. | 0 |
| [9:8] | CC2S[1:0] | RW | Compare capture channel 2 input selection fields.<br>00: Compare capture channel 2 is configured as an output.<br>01: Compare capture channel 2 is configured as an input and IC2 is mapped on TI2.<br>10: Compare capture channel 2 is configured as an input and IC2 is mapped on TI1.<br>11: Compare Capture Channel 2 is configured as an input and IC2 is mapped on TRC. This mode works only when the internal trigger input is selected (by the TS bit).<br>*Note: Compare Capture Channel 2 is writable only when the channel is off (When CC2E is zero).* | 0 |
| 7 | OC1CE | RW | Compare capture channel 1 clear enable bit. | 0 |
| [6:4] | OC1M[2:0] | RW | Compare capture channel 1 mode setting field. | 0 |
| 3 | OC1PE | RW | Compare capture register 1 preload enable bit. | 0 |
| 2 | OC1FE | RW | Compare capture channel 1 fast enable bit. | 0 |
| [1:0] | CC1S[2:0] | RW | Compare capture channel 1 input selection fields. | 0 |

Capture mode (pin direction is input).

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:12] | IC2F[3:0] | RW | The input capture filter 2 configuration field, these bits set the sampling frequency of the TI1 input and the digital filter length. The digital filter consists of an event counter, which records N events and then generates a jump in the output.<br>0000: No filter, sampled at fDTS.<br>1000: Sampling frequency Fsampling = Fdts/8, N = 6.<br>0001: Sampling frequency Fsampling=Fck_int, N=2.<br>1001: Sampling frequency Fsampling = Fdts/8, N = 8.<br>0010: Sampling frequency Fsampling=Fck_int, N=4.<br>1010: Sampling frequency Fsampling = Fdts/16, N = 5.<br>0011: Sampling frequency Fsampling=f=Fck_int, N=8.<br>1011: Sampling frequency Fsampling = Fdts/16, N = 6.<br>0100: Sampling frequency Fsampling = Fdts/2, N = 6. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1100: Sampling frequency Fsampling = Fdts/16, N = 8.<br>0101: Sampling frequency Fsampling = Fdts/2, N = 8.<br>1101: Sampling frequency Fsampling = Fdts/32, N = 5.<br>0110: Sampling frequency Fsampling = Fdts/4, N = 6.<br>1110: Sampling frequency Fsampling = Fdts/32, N = 6.<br>0111: Sampling frequency Fsampling = Fdts/4, N = 8.<br>1111: Sampling frequency Fsampling=Fdts/32, N=8. | |
| [11:10] | IC2PSC[1:0] | RW | Compare capture channel 2 prescaler configuration field, these 2 bits define the prescaler coefficient for compare capture channel 2. Once CC1E = 0, the prescaler is reset.<br>00: Without prescaler, one capture is triggered for each edge detected on the capture input.<br>01: Capture triggered every 2 events.<br>10: Capture triggered every 4 events.<br>11: Capture is triggered every 8 events. | 0 |
| [9:8] | CC2S[1:0] | RW | Compare the capture channel 2 input selection field, these 2 bits define the direction of the channel (Input/output), and the selection of the input pin.<br>00: Compare capture channel 1 channel is configured as an output.<br>01: Compare capture channel 1 channel is configured as an input and IC1 is mapped on TI1.<br>10: Compare capture channel 1 channel is configured as an input and IC1 is mapped on TI2.<br>11: Compare capture channel 1 channel is configured as an input and IC1 is mapped on TRC. This mode works only when the internal trigger input is selected (By the TS bit).<br>*Note: CC1S is writable only when the channel is off (CC1E is 0).* | 0 |
| [7:4] | IC1F[3:0] | RW | Input capture filter 1 configuration field. | 0 |
| [3:2] | IC1PSC[1:0] | RW | Compare capture channel 1 prescaler configuration field. | 0 |
| [1:0] | CC1S[1:0] | RW | Compare capture channel 1 input selection field. | 0 |

### 12.4.8 Compare/Capture Control Register 2 (TIMx_CHCTLR2) (x=1/2)

Offset address: 0x1C

The channel can be used in input (Capture mode) or output (Compare mode), and the direction of the channel is defined by the corresponding CCxS bit. The other bits of this register serve different purposes in input and output modes. OCxx describes the function of the channel in output mode and ICxx describes the function of the channel in input mode.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OC4CE | OC4M[2:0] | | | OC4PE | OC4FE | CC4S[1:0] | | OC3CE | OC3M[2:0] | | | OC3PE | OC3FE | CC3S[1:0] | |
| IC4F[3:0] | | | | IC4PSC[1:0] | | | | IC3F[3:0] | | | | IC3PSC[1:0] | | | |

Compare mode (Pin direction is output).

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | OC4CE | RW | Compare capture channel 4 clear enable bit. | 0 |
| [14:12] | OC4M[2:0] | RW | Compare capture channel 4 mode setting field. | 0 |
| 11 | OC4PE | RW | Compare capture register 4 preload enable bit. | 0 |
| 10 | OC4FE | RW | Compare capture channel 4 fast enable bit. | 0 |

| | | | | |
|---|---|---|---|---|
| [9:8] | CC4S[1:0] | RW | Compare capture channel 4 input selection fields. | 0 |
| 7 | OC3CE | RW | Compare capture channel 3 clear enable bit. | 0 |
| [6:4] | OC3M[2:0] | RW | Compare capture channel 3 mode setting field. | 0 |
| 3 | OC3PE | RW | Compare capture register 3 preload enable bit. | 0 |
| 2 | OC3FE | RW | Compare capture channel 3 fast enable bit. | 0 |
| [1:0] | CC3S[1:0] | RW | Compare capture channel 3 input selection fields. | 0 |

Capture mode (pin direction is input).

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:12] | IC4F[3:0] | RW | Input capture filter 4 configuration field. | 0 |
| [11:10] | IC4PSC[1:0] | RW | Compare capture channel 4 prescaler configuration field. | 0 |
| [9:8] | CC4S[1:0] | RW | Compare capture channel 4 input selection fields. | 0 |
| [7:4] | IC3F[3:0] | RW | Input capture filter 3 configuration field. | 0 |
| [3:2] | IC3PSC[1:0] | RW | Compare capture channel 3 prescaler configuration fields. | 0 |
| [1:0] | CC3S[1:0] | RW | Compare capture channel 3 input selection fields. | 0 |

### 12.4.9 Compare/Capture Enable Register 2 (TIMx_CCER) (x=1/2)

Offset address: 0x20

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | CC4P | CC4E | CC3NP | CC3NE | CC3P | CC3E | CC2NP | CC2NE | CC2P | CC2E | CC1NP | CC1NE | CC1P | CC1E |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:14] | Reserved | RO | Reserved | 0 |
| 13 | CC4P | RW | Compare capture channel 4 output polarity setting bit. | 0 |
| 12 | CC4E | RW | Compare capture channel 4 output enable bit. | 0 |
| 11 | CC3NP | RW | Compare capture channel 3 complementary output polarity setting bit. | 0 |
| 10 | CC3NE | RW | Compare capture channel 3 complementary output enable bits. | 0 |
| 9 | CC3P | RW | Compare capture channel 3 output polarity setting bit. | 0 |
| 8 | CC3E | RW | Compare capture channel 3 output enable bit. | 0 |
| 7 | CC2NP | RW | Compare capture channel 2 complementary output polarity setting bit. | 0 |
| 6 | CC2NE | RW | Compare capture channel 2 complementary output enable bits. | 0 |
| 5 | CC2P | RW | Compare capture channel 2 output polarity setting bit. | 0 |
| 4 | CC2E | RW | Compare capture channel 2 output enable bit. | 0 |
| 3 | CC1NP | RW | Compare capture channel 1 complementary output polarity setting bit. | 0 |
| 2 | CC1NE | RW | Compare capture channel 1 complementary output enable bit. | 0 |
| 1 | CC1P | RW | Compare capture channel 1 output polarity setting bit. | 0 |
| 0 | CC1E | RW | Compare capture channel 1 output enable bit. | 0 |

### 12.4.10 Counter for Advanced-control Timer (TIMx_CNT) (x=1/2)

Offset address: 0x24

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNT[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | CNT[15:0] | RW | The real-time value of the timer's counter. | 0 |

### 12.4.11 Counting Clock Prescaler (TIMx_PSC) (x=1/2)

Offset address: 0x28

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PSC[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | PSC[15:0] | RW | The dividing factor of the prescaler of the timer; the clock frequency of the counter is equal to the input frequency of the divider/(PSC+1). | 0 |

### 12.4.12 Auto-reload Value Register (TIMx_ATRLR) (x=1/2)

Offset address: 0x2C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ARR[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | ARR[15:0] | RW | The value of this field will be loaded into the counter, see section 12.2.3 for when the ATRLR acts and updates; the counter stops when the ATRLR is empty. | 0 |

### 12.4.13 Repeat Count Value Register (TIMx_RPTCR) (x=1/2)

Offset address: 0x30

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | REP [7:0] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved | 0 |
| [7:0] | REP | RW | The value of the repeat counter. | 0 |

### 12.4.14 Compare/Capture Register 1 (TIMx_CH1CVR) (x=1/2)

Offset address: 0x34

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | LEVEL1 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR1[15：0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:17] | Reserved | RO | Reserved | 0 |
| 16 | LEVEL1 | RO | Level indication bit corresponding to the captured | 0 |

| | | | value | |
|---|---|---|---|---|
| [15:0] | CCR1[15:0] | RW | Compare/capture the value of register channel 1. | 0 |

### 12.4.15 Compare/Capture Register 2 (TIMx_CH2CVR) (x=1/2)

Offset address: 0x38

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | LEVEL2 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR2[15：0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:17] | Reserved | RO | Reserved | 0 |
| 16 | LEVEL2 | RO | Level indication bit corresponding to the captured value | 0 |
| [15:0] | CCR2[15:0] | RW | Compare/capture the value of register channel 2. | 0 |

### 12.4.16 Compare/Capture Register 3 (TIMx_CH3CVR) (x=1/2)

Offset address: 0x3C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR3[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR3[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | CCR3 [15:0] | RW | Compare/capture the value of register channel 3. | 0 |

### 12.4.17 Compare/Capture Register 4 (TIMx_CH4CVR) (x=1/2)

Offset address: 0x40

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR4[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR4[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | CCR4 [15:0] | RW | Compare/capture the value of register channel 4. | 0 |

### 12.4.18 Brake and Dead-time Register (TIMx_BDTR) (x=1/2)

Offset address: 0x44

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MOE | AOE | BKP | BKE | OSSR | OSSI | LOCK[1:0] | | DTG[7:0] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | MOE | RW | Main output enable bit. Once the brake signal is active, it will be cleared asynchronously.<br>1: Enable OCx and OCxN to be set as outputs.<br>0: Disable the output of OCx and OCxN or force to idle state. | 0 |
| 14 | AOE | RW | Auto output enable.<br>1: MOE can be set by software or set in the next update event.<br>0: MOE can only be set by software. | 0 |
| 13 | BKP | RW | The brake input polarity setting bit.<br>1: Brake input active high.<br>0: Brake input active low.<br>*Note: When LOCK level 1 is set, this bit cannot be modified. A write to this bit requires an HB clock before it can take effect.* | 0 |
| 12 | BKE | RW | Brake function enable bit.<br>1: Brake input enabled.<br>0: Brake input disabled.<br>*Note: When LOCK level 1 is set, this bit cannot be modified. A write to this bit requires an HB clock before it can take effect.* | 0 |
| 11 | OSSR | RW | 1: When the timer is not operating, once CCxE=1 or CCxNE=1, first turn on OC/OCN and outputinvalid level, then set OCx, OCxN enable output signal=1.<br>0: When the timer is not operating, OC/OCN output is disabled.<br>*Note: When LOCK level 1 is set, this bit cannot be modified.* | 0 |
| 10 | OSSI | RW | 1: When the timer is not operating, once CCxE = 1 or CCxNE = 1, OC/OCN first outputs its idle level, then OCx, OCxN enable output signal = 1.<br>0: When the timer is not operating, OC/OCN output is disabled.<br>*Note: When LOCK level 1 is set, this bit cannot be modified.* | 0 |
| [9:8] | LOCK[1:0] | RW | Lock the function setting field.<br>00: Disable the locking function.<br>01: Lock level 1, no DTG, BKE, BKP, AOE, OISx and OISxN bits can be written.<br>10: Lock level 2, where the bits in lock level 1 cannot be written, nor the CC polarity bits, nor the OSSR and OSSI bits.<br>11: Lock level 3, cannot write to the bits in lock level 2, and cannot write to the CC control bits.<br>*Note: After system reset, the LOCK bit can only be written once and cannot be modified again until reset.* | 0 |

| | | | | |
|---|---|---|---|---|
| [7:0] | DTG[7:0] | RW | Dead-time setting bits that define the duration of the dead-time between complementary outputs.<br>Assume that DT denotes its duration.<br>DTG[7:5]=0xx=>DT=DTG[7:0]*Tdtg, Tdtg=TDTS;<br>DTG[7:5]=10x=>DT=(64+DTG[5:0])*Tdtg, Tdtg=2*TDTS;<br>DTG[7:5]=110=>DT=(32+DTG[4:0])*Tdtg, Tdtg =8×TDTS;<br>DTG[7:5]=111=>DT=(32+DTG[4:0])*Tdtg, Tdtg =16*TDTS.<br>2023.9.11<br>*Note: Once the LOCK level (LOCK[1:0] bits in the TIMx_BDTR register) is set to 01b, 10b, or 11b, these bits cannot be modified.* | 0 |

### 12.4.19 DMA Control Register (TIMx_DMACFGR) (x=1/2)

Offset address: 0x48

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | DBL[4:0] | | | | | Reserved | | | DBA[4:0] | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:13] | Reserved | RO | Reserved | 0 |
| [12:8] | DBL[4:0] | RW | The length of the DMA continuous transmission, the actual value of which is the value of this field + 1. | 0 |
| [7:5] | Reserved | RO | Reserved | 0 |
| [4:0] | DBA[4:0] | RW | These bits define the offset of the DMA in continuous mode from the address where control register 1 is located. | 0 |

### 12.4.20 DMA Address Register for Continuous Mode (TIMx_DMAADR) (x=1/2)

Offset address: 0x4C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DMAB[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | DMAB [15:0] | RW | The address of the DMA in continuous mode. | 0 |

### 12.4.21 SPEC Register (TIMx_SPEC) (x=1/2)

Offset address: 0x50

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOGGLE | Reserved | | | | | | | SPEC_PWM_OC4 | SPEC_PWM_OC3 | SPEC_PWM_OC2 | SPEC_PWM_OC1 | Reserved | | PWM_EN[1:0] | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | TOGGLE | RO | Valid channel indication bits:<br>1: The current channel output is an invalid level;<br>0: The current channel output is a valid level. | 0 |
| [14:8] | Reserved | RO | Reserved. | 0 |

| | | | | |
|---|---|---|---|---|
| 7 | PWM_OC4 | RW | In alternate enable mode, the channel 4 invalid level is configured as follows<br>1: Invalid level is high;<br>0: Invalid level is low. | 0 |
| 6 | PWM_OC3 | RW | In alternate enable mode, the channel 3 invalid level is configured as follows<br>1: Invalid level is high;<br>0: Invalid level is low. | 0 |
| 5 | PWM_OC2 | RW | In alternate enable mode, the channel 2 invalid level is configured as follows<br>1: Invalid level is high;<br>0: Invalid level is low. | 0 |
| 4 | PWM_OC1 | RW | In alternate enable mode, the channel 1 invalid level is configured as follows<br>1: Invalid level is high;<br>0: Invalid level is low. | 0 |
| [3:2] | Reserved | RO | Reserved. | 0 |
| 1 | PWM_EN | RW | Channel 3 and Channel 4 alternate enable function enable bits:<br>1: Enable the alternate enable function;<br>0: Disable the alternate enable function. | 0 |
| 0 | PWM_EN | RW | Channel 1 and Channel 2 alternate enable function enable bits:<br>1: Enable the alternate enable function;<br>0: Disable the alternate enable function. | 0 |

# Chapter 13 General-Purpose Timer (GPTM)

The general-purpose timer module contains a 16-bit auto-reloadable timer (TIM3), for measuring pulse width or generating pulses of a specific frequency, PWM waves, etc. It can be used in automation control, power supply, etc.

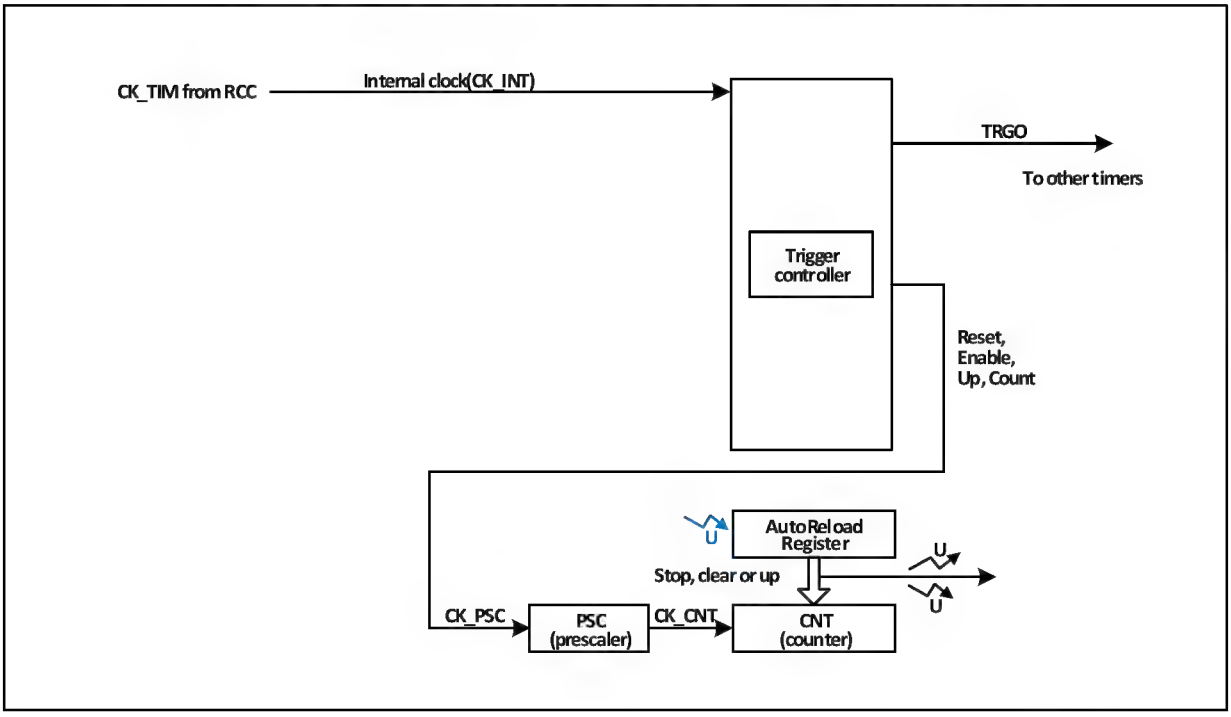
## 13.1 Main Features

The main features of the general-purpose timer include:

- 16-bit auto-reload counter, supports incremental counting mode
- 16-bit prescaler with dynamically adjustable crossover factor from 1 to 65536
- Support 2 independent comparison capture channels
- Each compare capture channel supports multiple operating modes, such as: input capture, output compare, PWM generation, and single pulse output
- Support external signal control timer

## 13.2 Principle and Structure

Figure 13-1 Block diagram of the structure of the general-purpose timer




### 13.2.1 Overview

As shown in Figure 13-1, the structure of the general-purpose timer can be roughly divided into three parts, namely the input clock part, the core counter part and the compare capture channel part.

The clock for the general-purpose timer can come from the HB bus clock (CK_INT), from the external clock input pin (TIMx_ETR), from other timers with clock output (ITRx), and from the input of the compare capture

channel (TIMx_CHx). These input clock signals become CK_PSC clocks after various set filtering and dividing operations, etc., and are output to the core counter section. In addition, these complex clock sources can also be output as TRGO to other peripherals such as timers and ADCs.

The core of the general-purpose timer is a 16-bit counter (CNT). cK_PSC is divided by a prescaler (PSC) to become cK_CNT and then finally fed to the CNT, which supports incremental counting mode, decremental counting mode, and incremental and decremental counting mode, and has an auto-reload register (ATRLR) to reload the initialization value for the CNT at the end of each counting cycle.

The general-purpose timer has four sets of compare capture channels, each of which can input pulses from exclusive pins or output waveforms to pins, i.e., the compare capture channels support both input and output modes. The input of each channel of the compare capture register supports filtering, dividing, edge detection, and other operations, and supports mutual triggering between channels, and can also provide clock for the core counter CNT. Each comparison capture channel has a set of comparison capture registers (CHxCVR) that support comparison with the main counter (CNT) to output pulses.

### 13.2.2 Differences between General-Purpose Timer and Advanced-Control Timer

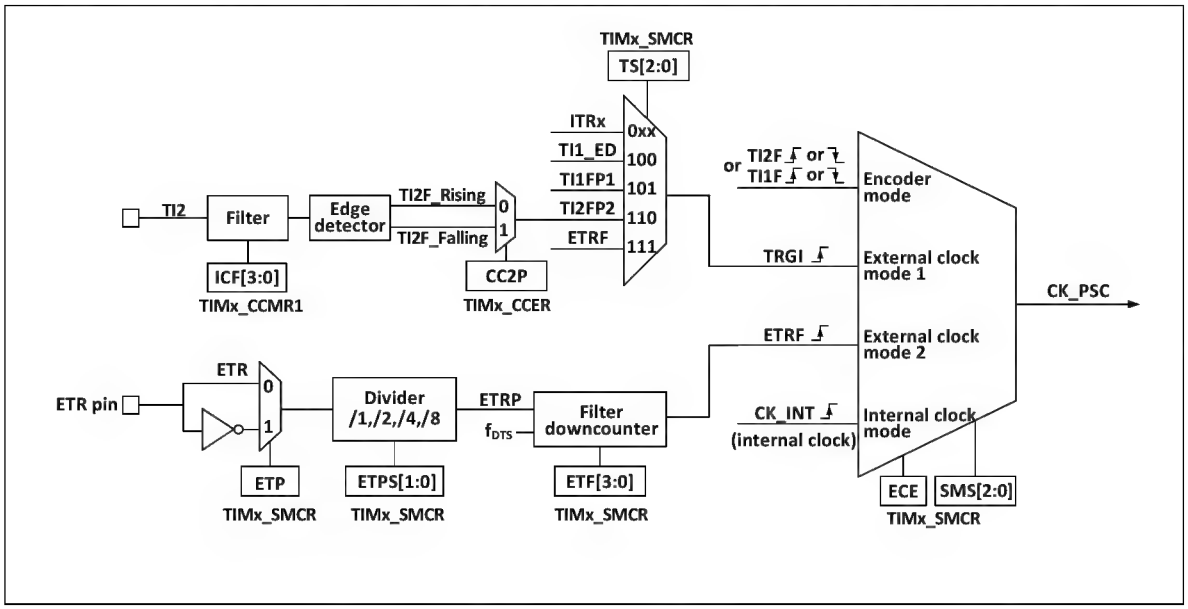
Compared to advanced-control timers, general-purpose timers lack the following features.

1) The general-purpose timer lacks a repeat count register for counting the count cycles of the core counter.
2) The comparison capture channel of the general-purpose timer lacks deadband generation and has no complementary output.
3) The general-purpose timer does not have a brake signal mechanism.
4) The general-purpose timer does not have encoder mode.

### 13.2.3 Clock Input

This section discusses the source of CK_PSC. The clock source portion of the overall block diagram of the general-purpose timer is captured here.

Figure 13-2 General-Purpose timer CK_PSC source block diagram




The optional input clocks can be divided into 4 categories:

1) Route of the external clock pin (ETR) input: ETR → ETRP → ETRF.

2) Internal HB clock input route: CK_INT.
3) Route from the compare capture channel pin (TIMx_CHx): TIMx_CHx → TIx → TIxFPx.
4) Input from other internal timers: ITRx.

The actual operation can be divided into 3 categories by determining the choice of input pulse for the SMS of the CK_PSC source.

1) Selection of the internal clock source (CK_INT).
2) External clock source mode 1.
3) External clock source mode 2.

All 3 clock source sources mentioned above can be selected by these 4 operations.

#### 13.2.3.1 Internal Clock Source (CK_INT)

If the general-purpose timer is started when the SMS field is held at 000b, then it is the internal clock source (CK_INT) that is selected as the clock. At this point CK_INT is CK_PSC.

#### 13.2.3.2 External Clock Source Mode 1

When the SMS domain is set to 111b, external clock source mode 1 is enabled. When external clock source 1 is enabled, TRGI is selected as the source for CK_PSC. it is worth noting that the user also needs to select the source for TRGI by configuring the TS domain. the TS domain can select the following types of pulses as clock sources.

1) Internal trigger (ITRx, x is 0,1,2,3).
2) Comparison of the signal after capturing channel 1 through the edge detector (TI1F_ED).
3) Comparison of signals TI1FP1, TI2FP2 of the capture channel.
4) The signal ETRF from the external clock pin input.

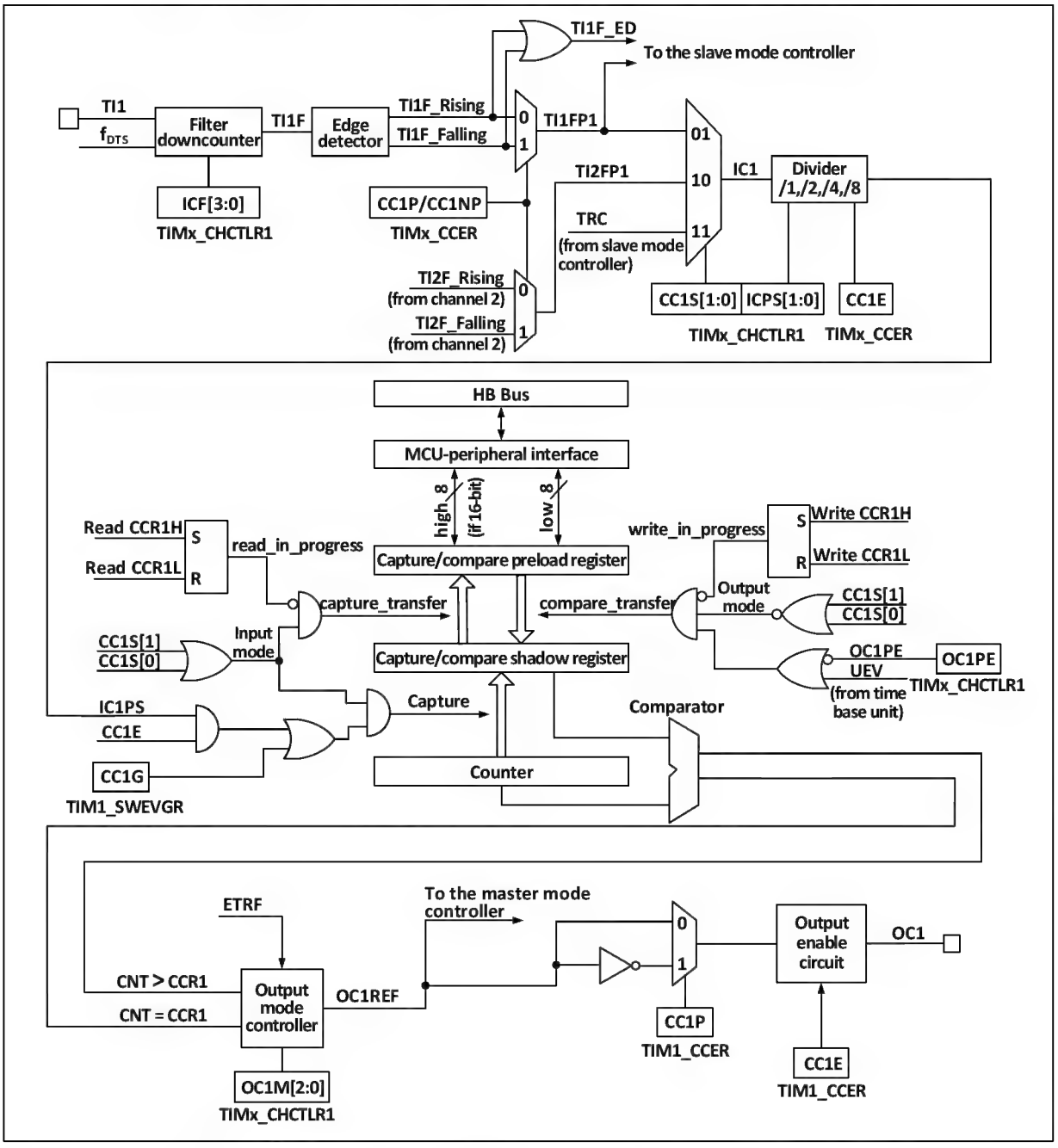
### 13.2.4 Counters and Peripherals

CK_PSC is input to the prescaler (PSC) for dividing. the PSC is 16-bit and the actual dividing factor is equal to the value of R16_TIMx_PSC + 1. CK_PSC goes through the PSC and becomes CK_INT. changing the value of R16_TIM1_PSC does not take effect in real time, but is updated to the PSC after an update event. the update event includes a UG bit clear and reset.

### 13.2.5 Compare/capture Channels

The core of the compare capture channel, which is the core of the timer to achieve complex functions, is the compare capture register, supplemented by digital filtering, frequency division and inter-channel multiplexing in the peripheral input section, and comparator and output control in the output section. The structure block diagram of the compare capture channel is shown in Figure 13-3.

Figure 13-3 Block diagram of the structure of the compare capture channel




The signal is input from the channel x pin and optionally made as TIx (The source of TI1 can be more than CH1, see block diagram 12-1 of the timer), TI1 is passed through the filter (ICF[3:0]) to generate TI1F, and then divided into TI1F_Rising and TI1F_Falling through the edge detector, these two signals are selected (CC1P) to generate TI1FP1, TI1FP1 and TI2FP1 from channel 2 are sent together to CC1S to select to become IC1, which is sent to the comparison capture register after ICPS dividing.

The compare capture register consists of a preload register and a shadow register, and the read/write process operates only on the preload register. In capture mode, the capture occurs on the shadow register and is then copied to the preload register; in compare mode, the contents of the preload register are copied to the shadow register, and then the contents of the shadow register are compared to the core counter (CNT).

## 13.3 Function and Implementation

The complex functions of a general-purpose timer are implemented by manipulating the timer's compare capture channel, clock input circuitry, and counter and peripheral components. The clock input to the timer can be derived from multiple clock sources including the input to the compare capture channel. The operation of the compare capture host channel and clock source selection directly determines its function. The compare capture channel is bidirectional and can operate in both input and output modes.

### 13.3.1 Counter Mode

**Incremental counting mode**

In incremental counting mode, the counter counts from 0 to the auto reload value (The contents of the R16_TIMx_ATRLR register), then counts again from 0 and generates a counter overflow event.

An update event will also be generated when setting the TIMx_SWEVGR register to UG position 1 (Either by software or using the slave mode controller).

The UEV event can be disabled by software by setting the UDIS position 1 in the R16_TIMx_CTLR1 register. This prevents the shadow register from being updated when a new value is written to the preload register. No update event will be generated until the UDIS bit is written to 0. However, both the counter and the prescaler counter will start counting from 0 again (While the prescaler ratio remains unchanged). In addition, if the URS bit (Update Request Select) in the R16_TIMx_CTLR1 register has been set to 1, setting UG location 1 will generate the update event UEV, but will not set the UIF flag to 1 (So no interrupts or DMA requests will be sent). This way, if the counter is cleared to zero when a capture event occurs, no update interrupt and no capture interrupt will be generated at the same time.

When an update event occurs, all registers will be updated and the update flag (UIF bit in the R16_TIMx_INTFR register) will be set to 1 (Depending on the URS bit):

1) The auto-reload shadow register will be updated with the preload value (R16_TIMx_ATRLR)
2) The prescaler buffer will be reloaded with the preload value (Contents of the R16_TIMx_PSC register)

### 13.3.2 Input Capture Mode

The input capture mode is one of the basic functions of the timer. The principle of input capture mode is that when a determined edge on the ICxPS signal is detected, a capture event is generated and the current value of the counter is latched into the compare capture register (R16_TIMx_CHCTLRx). The CCxIF (In R16_TIMx_INTFR) is set when a capture event occurs, and the corresponding interrupt or DMA is generated if enabled. If the CCxIF is already set when a capture event occurs, the CCxOF bit is set. the CCxIF can be cleared by software, or by hardware by reading the compare capture register. CCxOF is cleared by software.

An example of channel 1 to illustrate the steps to use the input capture mode is as follows.

1) Configure the CCxS domain to select the source of the ICx signal. For example, set it to 10b and select TI1FP1 as the source of IC1, not using the default setting, the CCxS domain defaults to making the comparison capture module the output channel.
2) Configure the ICxF domain to set the digital filter for the TI signal. The digital filter will sample the signal at a determined frequency, a determined number of times, and then output a hop. This sampling frequency and number of times is determined by ICxF.
3) Configure the CCxP bit to set the polarity of the TIxFPx. For example, keeping the CC1P bit low and selecting rising edge jumps.
4) Configure the ICxPS domain to set the ICx signal to be the crossover factor between ICxPS. For example,

keeping ICxPS at 00b, without crossover.

5) Configure the CCxE bit to allow capturing the value of the core counter (CNT) into the compare capture register. Set the CC1E bit.
6) Configure the CCxIE and CCxDE bits as needed to determine whether to allow enable interrupts or DMA.

This completes the comparison capture channel configuration.

When a captured pulse is input to TI1, the value of the core counter (CNT) is recorded in the compare capture register, CC1IF is set, and the CCIOF bit is set when CC1IF has been set before. If the CC1IE bit is set, then an interrupt is generated; if CC1DE is set, a DMA request is generated. An input capture event can be generated by software by way of writing the event generation register (R16_TIMx_SWEVGR).

### 13.3.3 Compare Output Mode

The compare output mode is one of the basic functions of the timer. The principle of the compare output mode is to output a specific change or waveform when the value of the core counter (CNT) agrees with the value of the compare capture register. the OCxM field (in R16_TIMx_CHCTLRx) and the CCxP bit (in R16_TIMx_CCER) determine whether the output is a definite high or low level or a level flip. The CCxIF bit is also set when a compare coherent event is generated. If the CCxIE bit is pre-set, an interrupt will be generated; if the CCxDE bit is pre-set, a DMA request will be generated.

To configure to compare output modes, proceed as follows.

1) Configure the clock source and auto-reload value of the core counter (CNT).
2) Set the count value to be compared to the comparison capture register (R16_TIMx_CHxCVR).
3) Set the CCxIE bit if an interrupt needs to be generated.
4) Keep OCxPE at 0 to disable the preload register for the compare capture register.
5) Setting the output mode, setting the OCxM field and the CCxP bit.
6) Enable the output, setting the CCxE bit.
7) Set the CEN bit to start the timer.

### 13.3.4 Forced Output Mode

The output pattern of the timer's compare capture channel can be forced by software to output a determined level without relying on comparison of the compare capture register's shadow register with the core counter.

This is done by setting OCxM to 100b, which forces OCxREF to low, or by setting OCxM to 101b, which forces OCxREF to high.

Note that by forcing OCxM to 100b or 101b, the comparison process between the internal main counter and the compare capture register is still going on, the corresponding flags are still set, and interrupts and DMA requests are still being generated.

### 13.3.5 PWM Input Mode

The PWM input mode is used to measure the duty cycle and frequency of PWM and is a special case of the input capture mode. The operation is the same as input capture mode except for the following differences: PWM occupies two compare capture channels and the input polarity of the two channels is set to opposite, one of the signals is set as trigger input and SMS is set to reset mode.

For example, to measure the period and frequency of the PWM wave input from TI1, the following operations are required.

1) Set TI1 (TI1FP1) to be the input of IC1 signal. Set CC1S to 01b.
2) Set TI1FP1 to rising edge active. Holding CC1P at 0.

3) Set TI1 (TI1FP2) as the input of IC2 signal. Set CC2S to 10b.
4) Select TI1FP2 to set to falling edge active. Set CC2P to 1.
5) Select TI1FP1 as the source of the clock source. set TS to 101b.
6) Set the SMS to reset mode, i.e. 100b.
7) Enables input capture. CC1E and CC2E are set.

*Note: Since only TI1FP1 and TI2FP2 are connected to the slave mode controller, only TIM3_CH1/TIM3_CH2 can be used in the PWM input mode.*

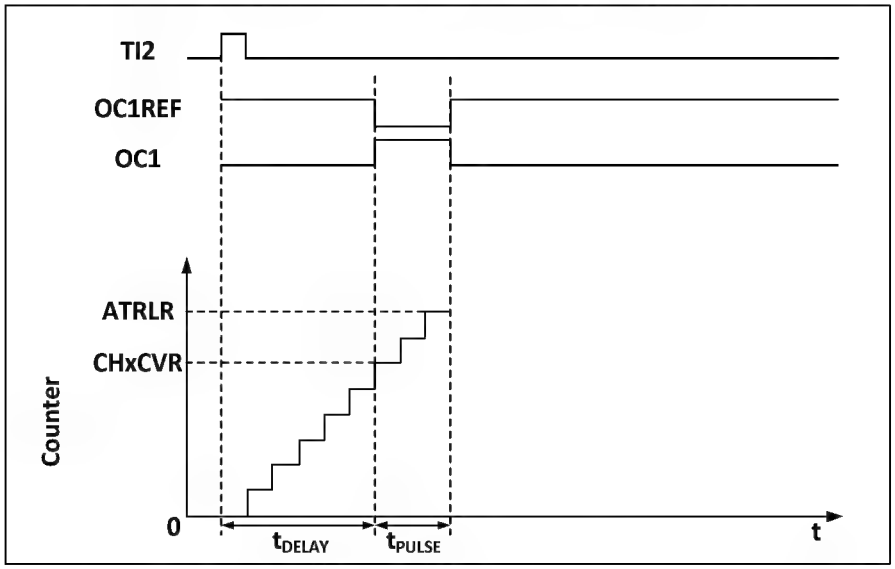
### 13.3.6 PWM Output Mode

PWM output mode is one of the basic functions of the timer. PWM output mode is most commonly used to determine the PWM frequency using the reload value and the duty cycle using the capture comparison register. Set 110b or 111b in the OCxM field to use PWM mode 1 or mode 2, set the OCxPE bit to enable the preload register, and finally set the ARPE bit to enable the automatic reload of the preload register. The value of the preload register can only be sent to the shadow register when an update event occurs, so the UG bit needs to be set to initialize all registers before the core counter starts counting.

### 13.3.7 Single Pulse Mode

The single pulse mode can respond to a specific event by generating a pulse after a delay, with programmable delay and pulse width. Setting the OPM bit stops the core counter when the next update event UEV is generated (Counter flips to 0).

Figure 13-4 Event generation and impulse response




As shown in Figure 13-4, a positive pulse of length Tpulse needs to be generated on OC1 after a delay Tdelay at the beginning of a rising edge detected on the TI2 input pin.

1) Set TI2 to trigger. Setting the CC2S field to 01b to map TI2FP2 to TI2; setting the CC2P bit to 0b to set TI2FP2 as rising edge detection; setting the TS field to 110b to set TI2FP2 as trigger source; setting the SMS field to 110b to set TI2FP2 to be used to start the counter.
2) Tdelay is defined by the Compare Capture Register and Tpulse is determined by the value of the Auto Reload Value Register and the Compare Capture Register.

### 13.3.8 Synchronization of TIMx Timers and External Triggers

The timer can be synchronized with an external trigger in reset mode, gated mode and trigger mode.

**Slave mode: reset mode**

The counter and its prescaler can be reinitialized in response to a trigger input event; if the URS bit of the R16_TIMx_CTLR1 register is low, an update event UEV is generated; all preload registers (R16_TIMx_ATRLR, R16_TIMx_CHxCVR) are then updated.

In the following example, the up counter is cleared to zero when a rising edge occurs on the TI1 input:

1) Configure channel 1 to detect the rising edge of TI1. Configure the input filter bandwidth (This example does not require any filter, so keep IC1F = 0000). There is no need to configure the capture divider as it is not used for trigger operation. the CC1S bit selects only the input capture source, i.e. CC1S=01 (In R16_TIMx_CCMR1). Write CC1P=0 and CC1NP='0' to the R16_TIMx_CCER register to verify polarity (Rising edge detection only).

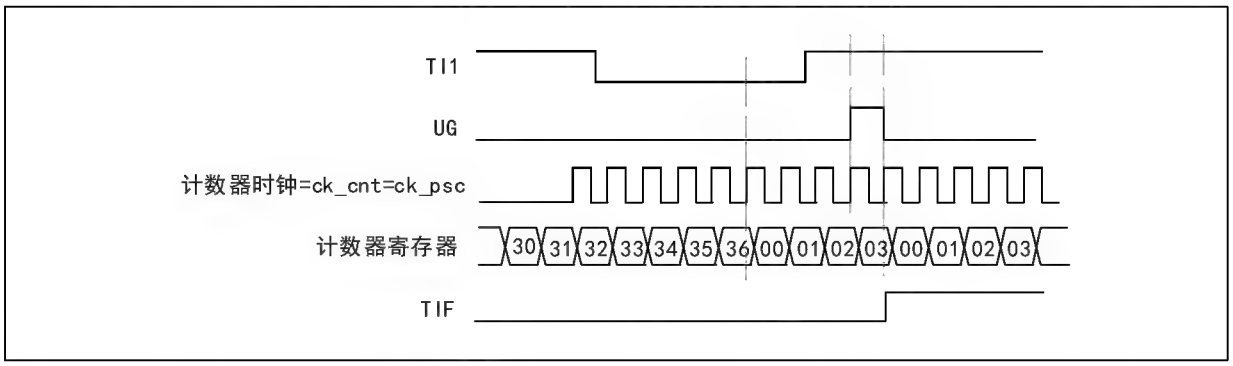
2) Write SMS=100 to R16_TIMx_SMCFGR to configure the timer to reset mode; write TS=101 to R16_TIMx_SMCFGR to select TI1 as input source.

3) Write CEN=1 to R16_TIMx_CTLR1 to start the counter.

The counter counts using the internal clock and then runs normally, when there is a TI1 rising edge the counter clears and starts counting again from 0. At the same time, the trigger flag TIF position 1, after enabling interrupt or DMA, allows an interrupt or DMA request to be sent. (Depends on the TIE (Interrupt Enable) bit and TDE (DMA Enable) bit in the R16_TIMx_DMAINTENR register).

The diagram below shows the action when the auto-reload register R16_TIMx_ARR = 0x36. The delay between the rising edge of TI1 and the actual counter reset is caused by the resynchronization circuitry at the TI1 input.

Figure 13-5 Control circuit in reset mode




**Slave mode: Gated mode**

The level of the input signal enables the counter. In the following example, the counter counts up only when TI1 is low:

1) Configure channel 1 to detect a low level on TI1. Configure the input filter bandwidth (This example does not require any filter, so keep IC1F = 0000). There is no need to configure the capture divider as it is not used for trigger operation. the CC1S bit selects only the input capture source, i.e. CC1S=01 (In R16_TIMx_CCMR1). Write CC1P=1 and CC1NP='0' to the R16_TIMx_CCER register to verify polarity (Detect low level only).

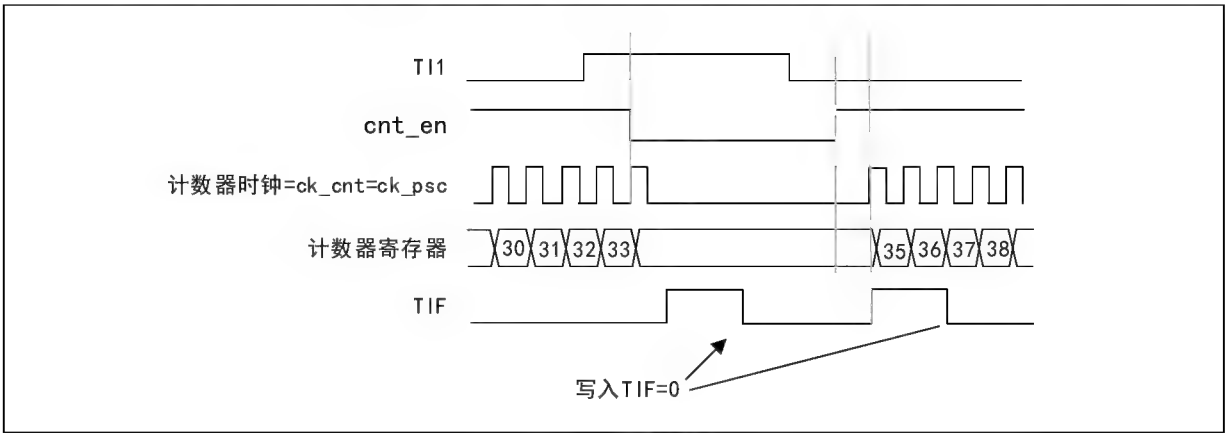
2) Write SMS=101 to R16_TIMx_SMCFGR to configure the timer for gated mode; write TS=101 to R16_TIMx_SMCFGR to select TI1 as input source.

3) Write CEN=1 to R16_TIMx_CTLR1 to start the counter. In gated mode, if CEN=0, the counter will not start regardless of the trigger input level.

As long as TI1 is low, the counter starts counting based on the internal clock and stops counting when TI1 goes high. When the counter starts or stops it sets TIF position 1 in R16_TIMx_INTFR. The delay between the rising

edge of TI1 and the actual counter reset is caused by the resynchronization circuitry at the TI1 input.

Figure 13-6 Control circuit in gated mode




**Slave mode: trigger mode**

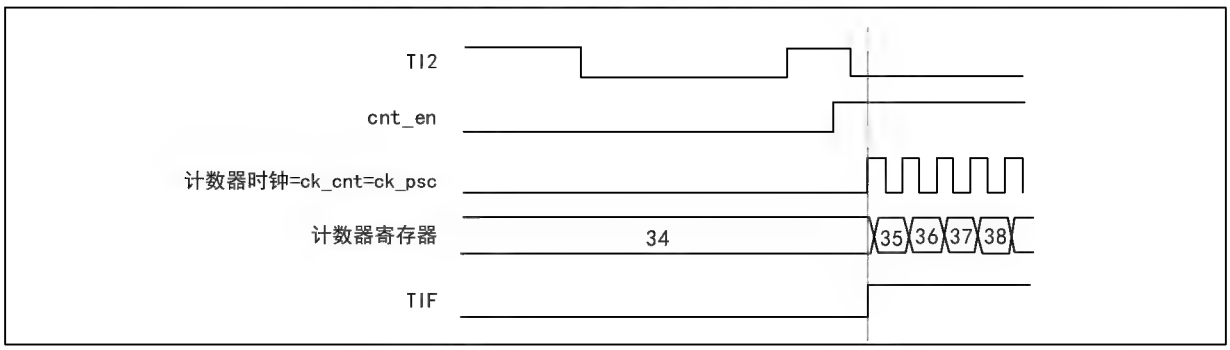
An event on the selected input will enable the counter. In the following example, the up counter is activated when a rising edge occurs on the TI2 input:

1) Configure channel 2 to detect the rising edge of TI2. Configure the input filter bandwidth (This example does not require any filter, so keep IC2F = 0000). There is no need to configure the capture divider as it is not used for trigger operation. the CC2S bit selects only the input capture source by setting CC2S=01 (In R16_TIMx_CCMR1). Write CC2P=1 and CC2NP='0' to the R16_TIMx_CCER register to verify polarity (Detect low level only)

2) Write SMS=110 to R16_TIMx_SMCFGR register to configure the timer to trigger mode; write TS=110 to R16_TIMx_SMCFGR register to select TI2 as input source.

When there is a rising edge of TI2, the counter starts counting driven by the internal clock, while TIF is set to 1. The delay between the rising edge of TI2 and the actual counter start is caused by the resynchronization circuitry at the TI2 input.

Figure 13-7 Control circuit in trigger mode




## 13.3.10 Debug mode

When the system enters the debug mode, the timer can be controlled to continue running or stop according to the setting of DBG module.

## 13.4 Register Description

Table 13-1 TIM3-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_TIM3_CTLR1 | 0x40000400 | TIM3 control register 1 | 0x0000 |
| R16_TIM3_SMCFGR | 0x40000408 | TIM3 slave mode control register | 0x0000 |
| R16_TIM3_DMAINTENR | 0x4000040C | TIM3 DMA/ interrupt enable register | 0x0000 |
| R16_TIM3_INTFR | 0x40000410 | TIM3 interrupt status register | 0x0000 |
| R16_TIM3_SWEVGR | 0x40000414 | TIM3 event generation register | 0x0000 |
| R16_TIM3_CHCTLR1 | 0x40000418 | TIM3 compare/capture control register1 | 0x0000 |
| R16_TIM3_CCER | 0x40000420 | TIM3 compare/capture enable register | 0x0000 |
| R16_TIM3_CNT | 0x40000424 | TIM3 counter | 0x0000 |
| R16_TIM3_PSC | 0x40000428 | TIM3 count clock prescaler | 0x0000 |
| R16_TIM3_ATRLR | 0x4000042C | TIM3 auto-reload value register | 0xFFFF |
| R16_TIM3_CH1CVR | 0x40000434 | TIM3 compare/capture register1 | 0x00000000 |
| R16_TIM3_CH2CVR | 0x40000438 | TIM3 compare/capture register2 | 0x00000000 |
| R16_TIM3_SPEC | 0x40000450 | SPEC register | 0x0000 |

### 13.4.1 Control Register 1 (TIM3_CTLR1)

Offset address: 0x00

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAPLVL | CAPOV | Reserved | | | | CKD[1:0] | | ARPE | Reserved | | | OPM | URS | UDIS | CEN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | CAPLVL | RW | In double-edge capture mode, the capture level indication is enabled.<br>1: Enable the indication function;<br>0: Disable the indication function.<br>*Note: When enabled, [16] of CHxCVR indicates the level corresponding to the capture value.* | 0 |
| 14 | CAPOV | RW | Capture value mode configuration.<br>1: The CHxCVR value is 0xFFFF when a counter overflow is generated before capture;<br>0: The capture value is the actual counter value. | 0 |
| [13:10] | Reserved | RO | Reserved | 0 |
| [9:8] | CKD[1:0] | RW | These 2 bits define the division ratio between the timer clock (CK_INT) frequency, the sampling clock used by the digital filter.<br>00: Tdts=Tck_int;<br>01: Tdts= 2xTck_int;<br>10: Tdts= 4xTck_int;<br>11： Reserved. | 0 |
| 7 | ARPE | RW | Auto-reload preload enable bit.<br>1: Enable the Auto-reload value register (ATRLR).<br>0: disabled the Auto-reload value register (ATRLR). | 0 |
| [6:4] | Reserved | RO | Reserved. | 0 |
| 3 | OPM | RW | Single pulse mode.<br>1: The counter stops when the next update event (Clearing the CEN bit) occurs.<br>0: The counter does not stop when the next update event occurs. | 0 |

| | | | | |
|---|---|---|---|---|
| 2 | URS | RW | Update request source, by which the software selects the source of the UEV event.<br>1: If an update interrupt or DMA request is enabled, only an update interrupt or DMA request is generated if the counter overflows/underflows.<br>0: If an update interrupt or DMA request is enabled, the update interrupt or DMA request is generated by any of the following events.<br>-Counter overflow/underflow<br>-Set the UG position<br>-Update generated from the mode controller | 0 |
| 1 | UDIS | RW | Disable updates, the software allows/disables the generation of UEV events via this bit.<br>1: UEV is disabled. no update event is generated and the registers (ATRLR, PSC, CHCTLRx) maintain their values. If the UG bitis set or a hardware reset is issued from the mode controller, the counter and prescaler are reinitialized.<br>0: UEV is allowed. update (UEV) events are generated by any of the following events:<br>- Counter overflow/underflow<br>-Set the UG position<br>-Update generated from the mode controller registers with caches are loaded with their preloaded values. | 0 |
| 0 | CEN | RW | Enable the counter (Counter enable).<br>1: Enable the counter.<br>0: Disable the counter.<br>*Note: The external clock, gated mode will not work until the CEN bit is set in software. Trigger mode can automatically set the CEN bit in hardware.* | 0 |

### 13.4.2 Slave Mode Control Register (TIM3_SMCFGR)

Offset address: 0x08

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | TS[2:0] | | | Reserved | SMS[2:0] | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:4] | TS[2:0] | RW | Trigger select field, these 3 bits select the trigger input source used to synchronize the counter.<br>000: Internal trigger 0 (ITR0).<br>001: Internal trigger 1 (ITR1).<br>010: Internal trigger 2 (ITR2).<br>011: Internal trigger 3 (ITR3).<br>100: Edge detector of TI1 (TI1F_ED).<br>101: Filtered timer input 1 (TI1FP1).<br>110: Filtered timer input 2 (TI2FP2).<br>111: External trigger input (ETRF).<br>The above only changes when SMS is 0. | 0 |
| 3 | Reserved | RO | Reserved. | 0 |
| [2:0] | SMS[2:0] | RW | Input mode selection field. Selects the clock and trigger mode of the core counter.<br>000: Driven by the internal clock CK_INT.<br>001: Reserved. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 010: Reserved.<br>011: Reserved.<br>100: Reset mode, where the rising edge of the trigger input (TRGI) will initialize the counter and generate a signal to update the registers.<br>101: Gated mode, when the trigger input (TRGI) is high, the counter clock is turned on; at the trigger input becomes low, the counter is stopped, and the counter starts and stops are controlled.<br>110: Trigger mode, where the counter is started on the rising edge of the trigger input TRGI and only the start of the counter is controlled.<br>111: External clock mode 1, rising edge of the selected trigger input (TRGI) drives the counter. | |

### 13.4.3 DMA/Interrupt Enable Register (TIM3_DMAINTENR)

Offset address: 0x0C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | TIE | Reserved | | | CC2IE | CC1IE | UIE |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| 6 | TIE | RW | Trigger interrupt enable bit.<br>1: Enable the trigger interrupt;<br>0: Disabled the trigger interrupt. | 0 |
| [5:3] | Reserved | RO | Reserved. | 0 |
| 2 | CC2IE | RW | Compare capture channel 2 interrupt enable bit.<br>1: Enable compare capture channel 2 interrupt;<br>0: Disable compare capture channel 2 interrupt. | 0 |
| 1 | CC1IE | RW | Compare capture channel 1 interrupt enable bit.<br>1: Enable compare capture channel 1 interrupt;<br>0: Disable compare capture channel 1 interrupt. | 0 |
| 0 | UIE | RW | Update interrupt enable bit.<br>1: Enable update interrupt;<br>0: Disabled update interrupt. | 0 |

### 13.4.4 Interrupt Status Register (TIM3_INTFR)

Offset address: 0x10

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | CC2OF | CC1OF | Reserved | | TIF | Reserved | | | CC2IF | CC1IF | UIF |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:11] | Reserved | RO | Reserved. | 0 |
| 10 | CC2OF | WO | Compare capture channel 2 over-capture flag bit. | 0 |
| 9 | CC1OF | WO | Compare capture channel 1 over-capture flag bit is only used when the compare capture channel is configured for input capture mode. This flag is set by hardware and a software write of 0 clears this bit.<br>1: The state of CC1IF has been set when the counter value is captured to the capture compare register;<br>0: No over-capture generated. | 0 |

| | | | | |
|---|---|---|---|---|
| [8:7] | Reserved | RO | Reserved. | 0 |
| 6 | TIF | WO | The trigger interrupt flag bit, which is set by hardware to this position when a trigger event occurs and cleared by software. Trigger events include the detection of a valid edge at the TRGI input when from a mode other than gated, or any edge in gated mode.<br>1: Trigger event generated;<br>0: No trigger event generated. | 0 |
| [5:3] | Reserved | RO | Reserved. | 0 |
| 2 | CC2IF | WO | Compare capture channel 2 interrupt flag bit. | 0 |
| 1 | CC1IF | WO | Compare capture channel 1 interrupt flag bit.<br>If the compare capture channel is configured in output mode, this bit is set by hardware when the counter value matches the compare value, except in centrosymmetric mode. This bit is cleared by software.<br>1: The value of the core counter matches the value of compare capture register 1; 0: no match occurs.<br>If compare capture channel 1 is configured in input mode, this bit is set by hardware when a capture event occurs and it is cleared by software or by reading the compare capture register.<br>1: The counter value has been captured compare capture register 1;<br>0: No input capture is generated. | 0 |
| 0 | UIF | WO | Update interrupt flag bit, this bit is set by hardware when an update event is generated and is cleared by software.<br>1: Update interrupt generated;<br>0: No update event is generated.<br>An update event will be generated in the following cases:<br>If UDIS=0, when the repeat counter value overflows or underflows;<br>If URS=0, UDIS=0, when the UG bit is set, or when the counter core counter is reinitialized by software;<br>If URS=0, UDIS=0, when the counter CNT is reinitialized by a trigger event; | 0 |

### 13.4.5 Event Generation Register (TIM3_SWEVGR)

Offset address: 0x14

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | TG | Reserved | | | CC2G | CC1G | UG |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved | 0 |
| 6 | TG | WO | The trigger event generation bit, which is set by software and cleared by hardware, is used to generate a trigger event.<br>1: Generate a trigger event, TIF is set, and the corresponding interrupts and DMAs are generated if enabled.<br>0: No action. | 0 |
| [5:3] | Reserved | RO | Reserved | 0 |

| | | | | |
|---|---|---|---|---|
| 2 | CC2G | WO | Compare capture event generation bit 2. Generate compare capture event 2. | 0 |
| 1 | CC1G | WO | Compare capture event generation bit 1. Generate compare capture event 1. This bit is set by software and cleared by hardware.<br>It is used to generate a compare capture event.<br>1: Generate a compare capture event on compare capture channel 1.<br>If compare capture channel 1 is configured as output: set the CC1IF bit. Generate the corresponding interrupts and DMAs if they are enabled.<br>If compare capture channel 1 is configured as input: the current core counter value is captured to compare capture register 1; set the CC1IF bit and generate the corresponding interrupts and DMAs if they are enabled. If CC1IF is already set, set the CC1OF bit.<br>0: No action. | 0 |
| 0 | UG | WO | Update event generation bit to generate an update event. This bit is set by software and is automatically cleared by hardware.<br>1: Initialize the counter and generate an update event.<br>0: No action.<br>*Note: The prescaler counter is also cleared to zero, but the prescaler factor remains unchanged. The core counter is cleared if in centrosymmetric mode or incremental counting mode; if in decremental counting mode, the core counter takes the value of the reload value register.* | 0 |

### 13.4.6 Compare/Capture Control Register 1 (TIM3_CHCTLR1)

Offset address: 0x18

The channel can be used in input (Capture mode) or output (Compare mode), and the direction of the channel is defined by the corresponding CCxS bit. The other bits of this register serve different purposes in input and output modes. OCxx describes the function of the channel in output mode and ICxx describes the function of the channel in input mode.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OC2CE | OC2M[2:0] | | | OC2PE | OC2FE | CC2S[1:0] | | OC1CE | OC1M[2:0] | | | OC1PE | OC1FE | CC1S[1:0] | |
| IC2F[3:0] | | | | IC2PSC[1:0] | | | | IC1F[3:0] | | | | IC1PSC[1:0] | | | |

Compare mode (pin direction is output).

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | OC2CE | RW | Compare capture channel 2 clear enable bit.<br>1: Clear the OC2REF bit zero once the ETRF input is detected high.<br>0: OC2REF is not affected by the ETRF input. | 0 |
| [14:12] | OC2M[2:0] | RW | Compare the Capture Channel 2 mode setting field.<br>The 3 bits define the action of the output reference signal OC2REF, which determines the values of OC2, OC2N. OC2REF is active high, while the active levels of OC2 and OC2N depend on the CC2P, CC2NP bits.<br>000: Freeze. Comparison of the value of the capture register with the value of the comparison between the | 0 |

| | | | | |
|---|---|---|---|---|
| | | | core counters does not work for OC1REF.<br>001: Force to set to valid level. Forcing OC1REF high when the core counter has the same value as the comparison capture register 2.<br>010: Force to set to invalid level. Forcing OC1REF low when the value of the core counter is the same as the comparison capture register 2.<br>011: Flip. Flips the level of OC1REF when the core counter is the same as the value of compare capture register 2.<br>100: Forced to invalid level. Forces OC1REF to low.<br>101: Force to valid level. Force OC1REF to high.<br>110: PWM mode 1: When counting up, once the core counter is less than the value of the comparison capture register, channel 2 is an effective level, otherwise it is an invalid level; When counting down, once the core counter is greater than the value of the comparison capture register, channel 2 is at an invalid level (OC2REF=0), otherwise it is at an effective level (Oc2Ref = 1);<br>111: PWM mode 2: When counting up, once the core counter is less than the value of the comparison capture register, channel 2 is at an invalid level, otherwise it is at an effective level; When counting down, once the core counter is greater than the value of the comparison capture register, Channel 2 is active (OC2REF=1), otherwise it is inactive (OC2REF=0).<br>*Note: This bit cannot be modified once the LOCK level is set to 3 and CC1S=00b. In PWM mode 1 or PWM mode 2, the OC1REF level is changed only when the comparison result is changed or when switching from freeze mode to PWM mode in the output comparison mode.* | |
| 11 | OC2PE | RW | Compare Capture Register 2 preload enable bit.<br>1: Turn on the preload function of the compare capture register 2, the read and write operations operate only on the preload register, and the preload value of the compare capture register 2 is loaded into the current shadow register when the update event comes.<br>0: Disable the preload function of compare capture register 2. The compare capture register 2 can be written at any time, and the newly written value takes effect immediately.<br>*Note: Once the LOCK level is set to 3 and CC2S=00, this bit cannot be modified. PWM mode can be used only in single pulse mode (OPM=1) without confirming the pre-load register, otherwise its action is not determined.* | 0 |
| 10 | OC2FE | RW | Compare Capture Channel 2 fast enable bit, this bit is used to speed up the response of the compare capture channel output to trigger input events.<br>1: The active edge of the input to the flipflop acts as if a comparison match has occurred. Therefore, the OC is set to the comparison level independent of the comparison result. The delay between the valid edge of the sample trigger and the output of the compare capture channel 2 is reduced to 3 clock cycles. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: Based on the value of the counter and compare capture register 1, compare capture channel 2 operates normally, even if the flip-flop is open. The minimum delay to activate the compare capture channel 2 output is 5 clock cycles when the input of the flipflop has a valid edge.<br>OC2FE only works when the channel is configured to PWM1 or PWM2 mode. | |
| [9:8] | CC2S[1:0] | RW | Compare capture channel 2 input selection fields.<br>00: comparison capture channel 2 is configured as an output.<br>01: comparison capture channel 2 is configured as an input and IC2 is mapped on TI2.<br>10: comparison capture channel 2 is configured as an input and IC2 is mapped onTI1.<br>11: Compare Capture Channel 2 is configured as an input and IC2 is mapped on TRC. This mode works only when the internal trigger input is selected (By the TS bit).<br>*Note: Compare capture channel 2 is writable only when the channel is off (When CC2E is zero).* | 0 |
| 7 | OC1CE | RW | Compare capture channel 1 clear enable bit. | 0 |
| [6:4] | OC1M[2:0] | RW | Compare capture channel 1 mode setting field. | 0 |
| 3 | OC1PE | RW | Compare capture register 1 preload enable bit. | 0 |
| 2 | OC1FE | RW | Compare capture channel 1 fast enable bit. | 0 |
| [1:0] | CC1S[1:0] | RW | Compare capture channel 1 input selection fields. | 0 |

Capture mode (pin direction is input).

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:12] | IC2F[3:0] | RW | The input capture filter 2 configuration field, these bits set the sampling frequency of the TI1 input and the digital filter length. The digital filter consists of an event counter, which records N events and then generates a jump in the output.<br>0000: no filter, sampled at fDTS.<br>1000: sampling frequency Fsampling = Fdts/8, N = 6.<br>0001: sampling frequency Fsampling=Fck_int, N=2.<br>1001: sampling frequency Fsampling = Fdts/8, N = 8.<br>0010: sampling frequency Fsampling=Fck_int, N=4.<br>1010: sampling frequency Fsampling = Fdts/16, N = 5.<br>0011: sampling frequency Fsampling=f=Fck_int, N=8.<br>1011: sampling frequency Fsampling = Fdts/16, N = 6.<br>0100: sampling frequency Fsampling = Fdts/2, N = 6.<br>1100: sampling frequency Fsampling = Fdts/16, N = 8.<br>0101: sampling frequency Fsampling = Fdts/2, N = 8.<br>1101: sampling frequency Fsampling = Fdts/32, N = 5.<br>0110: sampling frequency Fsampling = Fdts/4, N = 6.<br>1110: sampling frequency Fsampling = Fdts/32, N = 6.<br>0111: sampling frequency Fsampling = Fdts/4, N = 8.<br>1111: Sampling frequency Fsampling=Fdts/32, N=8. | 0 |
| [11:10] | IC2PSC[1:0] | RW | Compare capture channel 2 prescaler configuration field, these 2 bits define the prescaler coefficient for compare capture channel 2. Once CC1E = 0, the prescaler is reset.<br>00: without prescaler, one capture is triggered for each edge detected on the capture input.<br>01: capture triggered every 2 events. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 10: capture triggered every 4 events.<br>11: Capture is triggered every 8 events. | |
| [9:8] | CC2S[1:0] | RW | Compare the capture channel 2 input selection field, these 2 bits define the direction of the channel (Input/output), and the selection of the input pin.<br>00: Comparative capture channel 1 channel is configured as an output.<br>01: Comparison capture channel 1 channel is configured as an input and IC1 is mapped on TI1.<br>10: Comparison capture channel 1 channel is configured as an input and IC1 is mapped on TI2.<br>11: The compare capture channel 1 channel is configured as an input and IC1 is mapped on TRC. This mode works only when the internal trigger input is selected (By the TS bit).<br>*Note: CC1S is writable only when the channel is off (CC1E is 0).* | 0 |
| [7:4] | IC1F[3:0] | RW | Input capture filter 1 configuration field. | 0 |
| [3:2] | IC1PSC[1:0] | RW | Compare capture channel 1 prescaler configuration field. | 0 |
| [1:0] | CC1S[1:0] | RW | Compare capture channel 1 input selection fields. | 0 |

### 13.4.7 Compare/Capture Enable Register (TIM3_CCER)

Offset address: 0x20

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | CC2P | CC2E | Reserved | | CC1P | CC1E |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:6] | Reserved | RO | Reserved. | 0 |
| 5 | CC2P | RW | Compare capture channel 2 output polarity setting bit. | 0 |
| 4 | CC2E | RW | Compare capture channel 2 output enable bit. | 0 |
| [3:2] | Reserved | RO | Reserved. | 0 |
| 1 | CC1P | RW | Compare capture channel 1 output polarity setting bit. | 0 |
| 0 | CC1E | RW | Compare capture channel 1 output enable bit. | 0 |

### 13.4.8 Counter for General-purpose Timer (TIM3_CNT)

Offset address: 0x24

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNT[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | CNT[15:0] | RW | The real-time value of the timer's counter. | 0 |

### 13.4.9 Counting Clock Prescaler (TIM3_PSC)

Offset address: 0x28

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PSC[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| [15:0] | PSC[15:0] | RW | The dividing factor of the prescaler of the timer; the clock frequency of the counter is equal to the input frequency of the divider/(PSC+1). | 0 |

### 13.4.10 Auto-reload Value Register (TIM3_ATRLR)

Offset address: 0x2C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ARR[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | ARR[15:0] | RW | The value of ATRLR[15:0] will be loaded into the counter, read section 13.2.4 for when ATRLR acts and updates; the counter stops when ATRLR is empty. | 0xFFFF |

### 13.4.11 Compare/Capture Register 1 (TIM3_CH1CVR)

Offset address: 0x34

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | LEVEL1 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR1[15：0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:17] | Reserved | RO | Reserved | 0 |
| 16 | LEVEL1 | RO | Level indication bit corresponding to the captured value | 0 |
| [15:0] | CCR1[15:0] | RW | Compare/capture the value of register channel 1. | 0 |

### 13.4.12 Compare/Capture Register 2 (TIM3_CH2CVR)

Offset address: 0x34

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | LEVEL2 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR2[15：0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:17] | Reserved | RO | Reserved | 0 |
| 16 | LEVEL2 | RO | Level indication bit corresponding to the captured value | 0 |
| [15:0] | CCR2[15:0] | RW | Compare/capture the value of register channel 2. | 0 |

### 13.4.13 SPEC Register (TIM3_SPEC)

Offset address: 0x50

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOGGLE | Reserved | | | | | | | | | PWM_OC2 | PWM_OC1 | Reserved | | | PWM_EN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | TOGGLE | RO | Valid channel indication bits:<br>1: The current channel output is an invalid level;<br>0: The current channel output is a valid level. | 0 |
| [14:6] | Reserved | RO | Reserved. | 0 |
| 5 | PWM_OC2 | RW | In alternate enable mode, the channel 2 invalid level is configured as follows<br>1: Invalid level is high;<br>0: Invalid level is low. | 0 |
| 4 | PWM_OC1 | RW | In alternate enable mode, the channel 1 invalid level is configured as follows<br>1: Invalid level is high;<br>0: Invalid level is low. | 0 |
| [3:1] | Reserved | RO | Reserved. | 0 |
| 0 | PWM_EN | RW | Channel 1 and Channel 2 alternate enable function enable bits:<br>1: Enable the alternate enable function;<br>0: Disable the alternate enable function. | 0 |

# Chapter 14 Universal Synchronous Asynchronous Receiver Transmitter (USART)

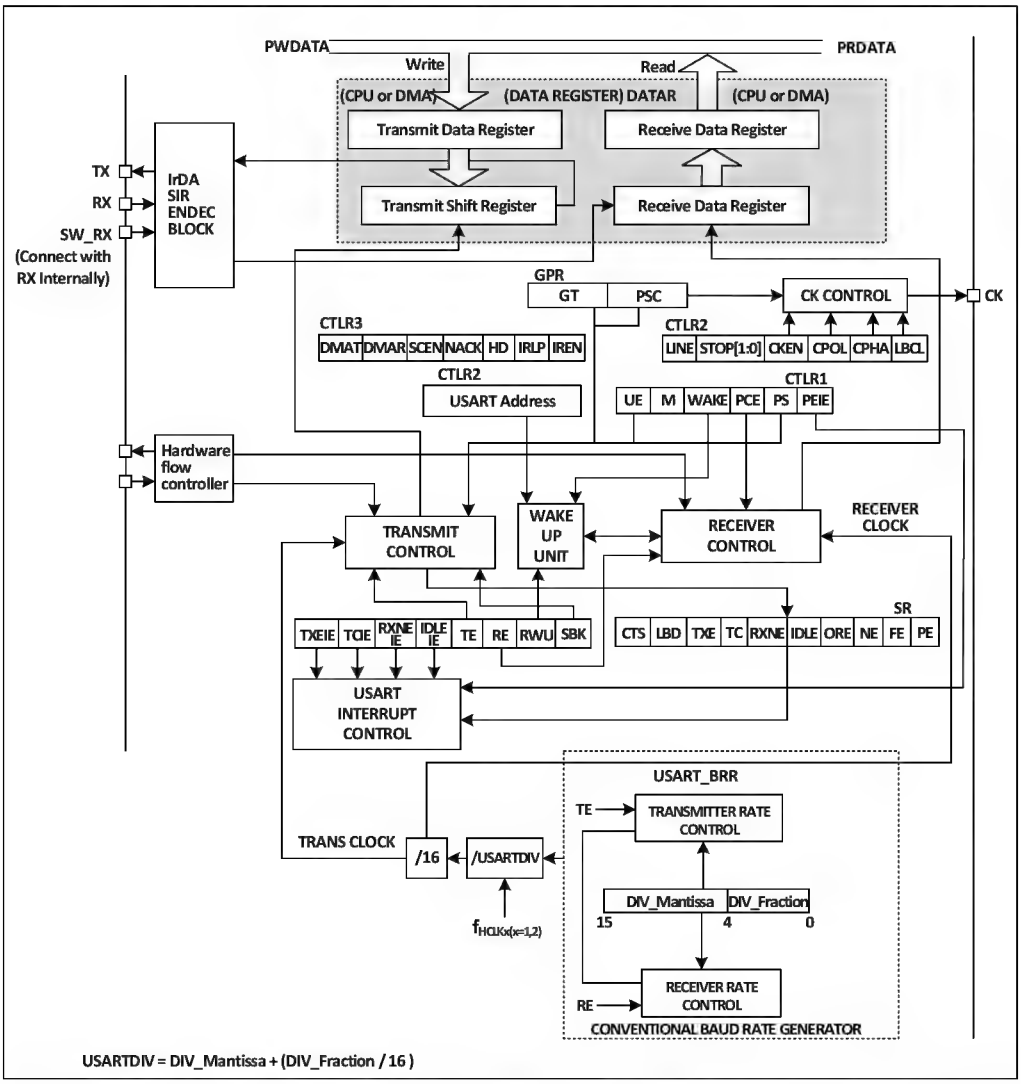
The module contains 4 Universal Synchronous Asynchronous Transceiver (USART1/2/3/4).

## 14.1 Main Features

- Full-duplex or half-duplex synchronous or asynchronous communication
- NRZ data format
- Fractional baud rate generator, up to 3Mbps
- Programmable data length
- Configurable stop bits
- Support LIN, IrDA encoders, smart cards
- Support DMA
- Multiple interrupt sources

## 14.2 Overview

Figure 14-1 Block diagram of a general-purpose synchronous/asynchronous transceiver




When TE (Transmit enable bit) is set, the data in the transmit shift register is output on the TX pin and the clock is output on the CK pin. When transmitting, the first bit shifted out is the least significant bit and each data frame starts with a low start bit, then the transmitter sends an 8- or 9-bit data word depending on the setting on the M (Word length) bit, and finally a configurable number of stop bits. If equipped with a parity check bit, the last bit of the data word is the check bit. After the TE is set an idle frame is sent, which is 10 or 11 bits high and contains the stop bit. The disconnect frame is 10 or 11 bits low followed by the stop bit.

## 14.3 Baud Rate Generator

The baud rate of the transceiver = HCLK/(16*USARTDIV), HCLK is the clock of HB. The value of USARTDIV is determined by the two fields DIV_M and DIV_F in USART_BRR, which is calculated by the formula. The formula is as follows.

$$USARTDIV = DIV\_M+(DIV\_F/16)$$

It should be noted that the baud rate generated by the baud rate generator may not be just what the user needs, which may be biased. In addition to taking as close a value as possible, a way to reduce the deviation is to increase

the HB clock. For example, if you set the baud rate to 115200bps, the value of USARTDIV is set to 39.0625, which will give you a baud rate of exactly 115200bps at the highest frequency, but if you need a baud rate of 921600bps, the calculated USARTDIV is 4.88, but the closest value filled in USART_BRR is actually only 4.875, the actual baud rate is 923076bps, which is 0.16% error.

When the serial waveform sent by the sender is transmitted to the receiver, the baud rate of the receiver and the sender is subject to some error. The error mainly comes from three aspects: the actual baud rate of the receiver and the sender is not the same; the receiver and the sender's clock has errors; the waveform in the line generated by the change. Peripheral module receiver is a certain receiving tolerance, when the sum of the above three aspects of the total deviation is less than the module's tolerance limit, the total deviation does not affect the transmission and reception. The tolerance limit of the module is affected by whether to use fractional baud rate and M-bit (Data field word length), using fractional baud rate and using 9-bit data field length will reduce the tolerance limit, but not less than 3%.

## 14.4 Synchronous Mode

Synchronous mode allows the system to output a clock signal when using the USART module. When synchronous mode is enabled to send data externally, the CK pin will output the clock externally at the same time.

The way to turn on the synchronous mode is to the CLKEN position bit in control register 2 (R16_USARTx_CTLR2), but also need to turn off the LIN mode, smart card mode, infrared mode and half duplex mode, i.e. ensure that the SCEN, HDSEL and IREN bits are in reset, these three in control register 3 (R16_USARTx_CTLR3).

The key point of using synchronous mode is the clock output control. There are several points to note.

The USART module synchronization mode works only in the main mode, i.e. the CK pin outputs only the clock and does not receive inputs.

Outputs a clock signal only when data is output on the TX pin.

The LBCL bit determines whether the clock is output when the last data bit is sent, the CPOL bit determines the polarity of the clock, and the CPHA determines the phase of the clock. These three bits are in control register 2 (R16_USARTx_CTLR2), which needs to be set when TE and RE are not enabled, see Figure 14-2 for the differences.

The receiver will only sample at the output clock in synchronous mode, requiring a certain amount of signal build time and hold time from the device, as shown in Figure 14-3.

Figure 14-2 USART clock timing example (M=0)

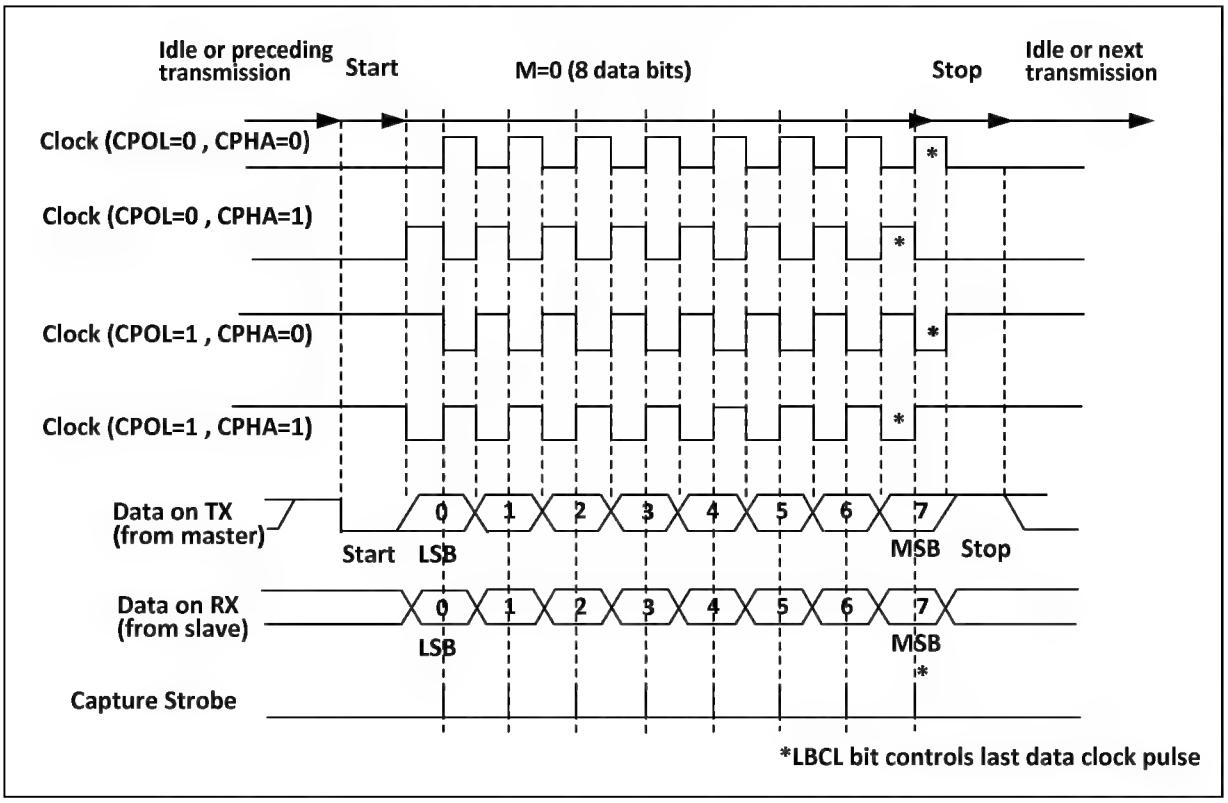
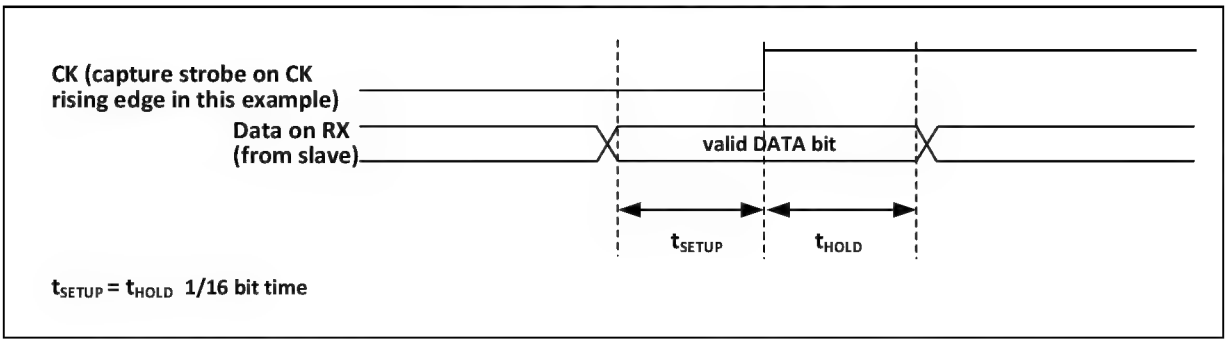



Figure 14-3 Data sample hold time


CK (capture strobe on CK rising edge in this example)
Data on RX (from slave)
valid DATA bit
tSETUP
tHOLD
tSETUP = tHOLD 1/16 bit time

## 14.5 1-Wire Half-Duplex Mode

Half-duplex mode supports the use of a single pin (TX pin only) for receive and transmit, with the TX and RX pins connected internally on the chip.

The way to turn on the half-duplex mode is to set the HDSEL position bit in control register 3 (R16_USARTx_CTLR3), but it is also necessary to turn off the LIN mode, smart card mode, IR mode and synchronous mode, i.e. to ensure that the SCEN, CLKEN and IREN bits are in reset, which are in control registers 2 and 3 (R16_USARTx_CTLR2 and R16_USARTx_CTLR3).

After setting to half-duplex mode, you need to set the IO port of the TX to output mode plus pull. With TE set, data will be sent out as soon as it is written to the data register. In particular, note that half-duplex mode may cause bus conflicts when multiple devices use a single bus to send and receive, which needs to be avoided by the user

with the software itself.

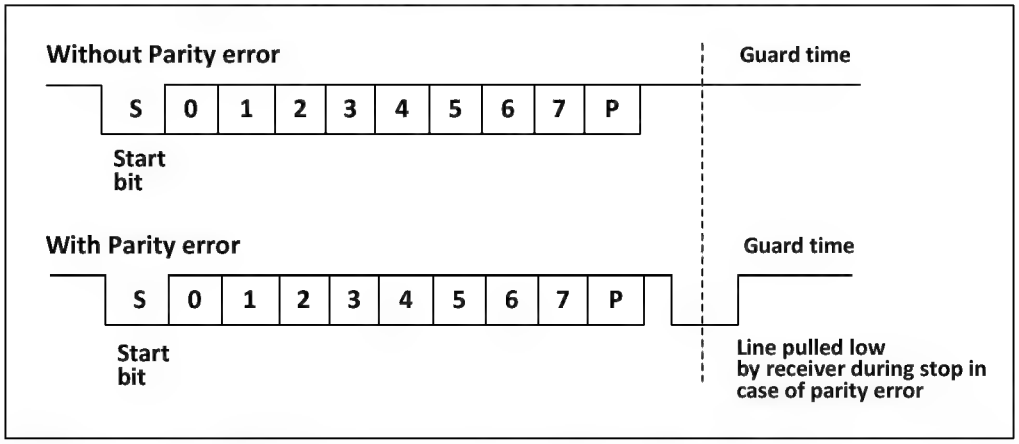
## 14.6 Smart Card

Smart card mode supports ISO7816-3 protocol access to smart card controllers.

The smart card mode is turned on by setting the SCEN position bit in control register 3 (R16_USARTx_CTLR3), but it is also necessary to turn off LIN mode, half duplex mode and IR mode, i.e. to ensure that the LINEN, HDSEL and IREN bits are in reset, but CLKEN can be turned on to output the clock, these bits are in control registers 2 and 3 (R16_USARTx_CTLR2 and R16_USARTx_CTLR3).

To support smart card mode, USART should be set to 8 bits of data plus 1 bit of parity, and its stop bit is recommended to be configured to 1.5 bits for both transmit and receive. Smart card mode is a single-wire half-duplex protocol that uses the TX line for data communication and should be configured as output pull-up. When the receiver receives a frame of data and detects a parity error, it sends a NACK signal, i.e., it actively pulls the TX down by one cycle during the stop bit, and the sender detects the NACK signal, which generates a frame error whereby the application can retransmit. Figure 14-4 shows the waveforms on the TX pin in the correct case and in the case of a parity error. the TC flag (Transmit complete flag) of the USART can delay the GT (Protection time) generation by one clock, and the receiver will not recognize the NACK signal it sets as the start bit.

Figure 14-4 (Un)Occurrence of parity error diagram




In smart card mode, the waveform output from the CK pin when enabled has nothing to do with communication; it simply clocks the smart card with the value of the HB clock followed by a five-bit settable clock division (Twice the value of the PSC, up to 62 divisions).

## 14.7 IrDA

The USART module supports control of IrDA infrared transceivers for physical layer communication. The LINEN, STOP, CLKEN, SCEN and HDSEL bits must be cleared to use IrDA. NRZ (Non-return to zero) coding is used between the USART module and the SIR physical layer (Infrared transceiver) and is supported up to 115200 bps rates.

IrDA is a half-duplex protocol, if UASRT is sending data to SIR physical layer, then IrDA decoder will ignore the newly sent IR signal, if USART is receiving data from SIR, then SIR will not accept the signal from USART. the level logic of USART to SIR and SIR to USART is different. In SIR receive logic, the high level is 1 and the low level is 0, but in SIR send logic, the high level is 0 and the low level is 1.

## 14.8 DMA

The USART module supports DMA function, which can be used to achieve fast and continuous transmitting and receiving. When DMA is enabled, the DMA writes data from the set memory space to the transmit buffer when TXE is set. When using DMA to receive, each time RXNE is set, DMA transfers the data in the receive buffer to a specific memory space.

## 14.9 Interrupts

The USART module supports a variety of interrupt sources, including transmit data register empty (TXE), CTS, transmit complete (TC), receive data ready (RXNE), data overflow (ORE), line idle (IDLE), parity error (PE), disconnect flag (LBD), noise (NE), overflow for multi-buffered communication (ORT), and frame error (FE), among others.

Table 14-1 Relationship between interrupts and corresponding enable bits

| Interrupt source | Enable bit |
|---|---|
| Transmit data register empty (TXE) | TXEIE |
| CTS interrupt (CTS) | CTSIE |
| Transmission complete (TC) | TCIE |
| Receive data register not empty (data ready to be read) | TXNEIE |
| Overrun error detected (ORE) | |
| Idle line detected (IDLE) | IDLEIE |
| Parity error (PE) | PEIE |
| LIN break (LBD) | LBDIE |
| Noise flag（NE） | EIE |
| Overflow of multi-buffered communication (ORT) | |
| Frame error (FE) for multibuffered communication | |

## 14.10 Register Description

Table 14-2 USART1-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_USART1_STATR | 0x40013800 | USART1 status register | 0x000000C0 |
| R32_USART1_DATAR | 0x40013804 | USART1 data register | 0x000000XX |
| R32_USART1_BRR | 0x40013808 | USART1 baud rate register | 0x00000000 |
| R32_USART1_CTLR1 | 0x4001380C | USART1 control register 1 | 0x00000000 |
| R32_USART1_CTLR2 | 0x40013810 | USART1 control register 2 | 0x00000000 |
| R32_USART1_CTLR3 | 0x40013814 | USART1 control register 3 | 0x00000000 |
| R32_USART1_GPR | 0x40013818 | USART1 guard time and prescaler register | 0x00000000 |

Table 14-3 USART2-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_USART2_STATR | 0x40004400 | USART2 status register | 0x000000C0 |
| R32_USART2_DATAR | 0x40004404 | USART2 data register | 0x000000XX |
| R32_USART2_BRR | 0x40004408 | USART2 baud rate register | 0x00000000 |
| R32_USART2_CTLR1 | 0x4000440C | USART2 control register 1 | 0x00000000 |

| | | | |
|---|---|---|---|
| R32_USART2_CTLR2 | 0x40004410 | USART2 control register 2 | 0x00000000 |
| R32_USART2_CTLR3 | 0x40004414 | USART2 control register 3 | 0x00000000 |
| R32_USART2_GPR | 0x40004418 | USART2 guard time and prescaler registers | 0x00000000 |

Table 14-4 USART3-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_USART3_STATR | 0x40004800 | USART3 status register | 0x000000C0 |
| R32_USART3_DATAR | 0x40004804 | USART3 data register | 0x000000XX |
| R32_USART3_BRR | 0x40004808 | USART3 baud rate register | 0x00000000 |
| R32_USART3_CTLR1 | 0x4000480C | USART3 control register 1 | 0x00000000 |
| R32_USART3_CTLR2 | 0x40004810 | USART3 control register 2 | 0x00000000 |
| R32_USART3_CTLR3 | 0x40004814 | USART3 control register 3 | 0x00000000 |
| R32_USART3_GPR | 0x40004818 | USART3 guard time and prescaler registers | 0x00000000 |

Table 14-4 USART4-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_USART4_STATR | 0x40004C00 | USART4 status register | 0x000000C0 |
| R32_USART4_DATAR | 0x40004C04 | USART4 data register | 0x000000XX |
| R32_USART4_BRR | 0x40004C08 | USART4 baud rate register | 0x00000000 |
| R32_USART4_CTLR1 | 0x40004C0C | USART4 control register 1 | 0x00000000 |
| R32_USART4_CTLR2 | 0x40004C10 | USART4 control register 2 | 0x00000000 |
| R32_USART4_CTLR3 | 0x40004C14 | USART4 control register 3 | 0x00000000 |
| R32_USART4_GPR | 0x40004C18 | USART4 guard time and prescaler registers | 0x00000000 |

### 14.10.1 USART Status Register (USARTx_STATR) (x=1/2/3/4)

Offset address: 0x00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | CTS | LBD | TXE | TC | RXNE | IDLE | ORE | NE | FE | PE |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:10] | Reserved | RO | Reserved | 0 |
| 9 | CTS | RW0 | CTS state change flag. If the CTSE bit is set, this bit will be set high by hardware when the nCTS output state changes. It is cleared to zero by software. If the CTSIE bit is already set, an interrupt will be generated.<br>1: There is a change on the nCTS status line;<br>0: There is no change on the nCTS status line. | 0 |
| 8 | LBD | RW0 | LIN Break detection flag. This bit is set by hardware when LIN Break is detected. Cleared by software. If LBDIE has been set, an interrupt will be generated.<br>1: LIN Break is detected;<br>0: LIN Break is not detected. | 0 |
| 7 | TXE | RO | Transmit data register empty flag. This bit is set by hardware when the data in the TDR register is | 1 |

| | | | | |
|---|---|---|---|---|
| | | | transferred to the shift register by hardware. If TXEIE has been set, an interrupt will be generated and the data register will be written and this bit will be reset.<br>1: The data has been transferred to the shift register;<br>0: The data has not been transferred to the shift register. | |
| 6 | TC | RW0 | Transmit completion flag. When a frame containing data is transmitted and TXE is set, the hardware will set this bit. If TCIE is set, a corresponding interrupt will be generated. The software will clear this bit after reading this bit and writing to the data register. You can also directly write 0 to clear this bit.<br>1: The transmitting is completed;<br>0: The transmitting is not yet completed. | 1 |
| 5 | RXNE | RW0 | Read the data register non-empty flag. When the data in the shift register is transferred to the data register, this bit will be set by hardware. If RXNEIE has been set, the corresponding interrupt will also be generated. Reading the data register clears this bit. You can also write 0 directly to clear this bit.<br>1: The data has been received and can be read;<br>0: The data has not been received yet. | 0 |
| 4 | IDLE | RO | Bus idle flag. This bit will be set by hardware when the bus is free. If IDLEIE has been set, the corresponding interrupt will be generated. Reading the status register and then the data register clears this bit.<br>1: Bus is idle;<br>0: Bus idle is not detected.<br>*Note: This bit will not be set again until RXNE is set.* | 0 |
| 3 | ORE | RO | Overload error flag. This bit will be set when there is data in the receive shift register that needs to be transferred to the data register, but there is still data in the receive field of the data register that has not been read out. If RXNEIE is set, the corresponding interrupt will also be generated.<br>1: An overload error occurs;<br>0: There is no overload error.<br>*Note: In case of an overload error, the value of the data register is not lost, but the value of the shift register is overwritten. If the EIE able bit is set, the ORE flag position bit generates an interrupt in multi-buffer communication mode.* | 0 |
| 2 | NE | RO | Noise error flag. It is set by hardware when the noise error flag is detected. The operation of reading the status register and then reading the data register resets this bit.<br>1: Noise detected.<br>0: No noise is detected.<br>*Note: This bit does not generate an interrupt. If the EIE bit is set, the FE flag position bit generates an interrupt in multi-buffer communication mode.* | 0 |
| 1 | FE | RO | Frame error flag. This bit will be set by hardware when a synchronization error, excessive noise or disconnect character is detected. Reading this bit | 0 |

| | | | | |
|---|---|---|---|---|
| | | | and then reading the data register operation will reset this bit.<br>1: Frame error detected.<br>0: No frame error detected.<br>*Note: This bit will not generate an interrupt. If the EIE bit is set, the FE flag position bit will generate an interrupt in multi-buffer communication mode.* | |
| 0 | PE | RO | Checksum error flag. In receive mode, hardware sets this bit if a parity check error is generated. A read of this bit and then a read of the data register operation resets this bit. Before clearing this bit, software must wait for the RXNE flag bit to be set. If the PEIE has been set previously, then this bit being set generates a corresponding interrupt.<br>1: A parity error.<br>0: No inspection error. | 0 |

### 14.10.2 USART Data Register (USARTx_DATAR) (x=1/2/3/4)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | | DR[8:0] | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:9] | Reserved | RO | Reserved | 0 |
| [8:0] | DR[8:0] | RW | Data register. This register is actually the receive data register (RDR) and send register (TDR) two registers composed of DR read and write operation start is read receive register (RDR) and write send register (TDR) respectively. | X |

### 14.10.3 USART Baud Rate Register (USARTx_BRR) (x=1/2/3/4)

Offset address: 0x08

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| DIV_Mantissa[11:0] | | | | | | | | | | | | DIV_Fraction[3:0] | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0 |
| [15:4] | DIV_Mantissa[11:0] | RW | These 12 bits define the integer part of the dividing factor of the frequency divider. | 0 |
| [3:0] | DIV_Fraction[3:0] | RW | These 4 bits define the fractional part of the dividing factor of the frequency divider. | 0 |

### 14.10.4 USART Control Register 1 (USARTx_CTLR1) (x=1/2/3/4)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | UE | M | WAKE | PCE | PS | PEIE | TXEIE | TCIE | RXNEIE | IDLEIE | TE | RE | RWU | SBK |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:14] | Reserved | RO | Reserved | 0 |
| 13 | UE | RW | USART enable bit. When this bit is cleared, both the USART divider and the output stop working after the current byte transfer is completed. | 0 |
| 12 | M | RW | Word long bit.<br>1: 9 data bits; 0: 8 data bits. | 0 |
| 11 | WAKE | RW | Wake-up bit. This bit determines the method of waking up the USART.<br>1: Address marker; 0: Bus idle. | 0 |
| 10 | PCE | RW | The parity bit is enabled. For the receiver, it is the parity check of the data; for the sender, it is the insertion of the parity bit. Once this bit is set, the parity bit enable will take effect only after the current byte transmission is completed. | 0 |
| 9 | PS | RW | Parity selection. 0 means even parity, 1 means odd parity. When this bit is set, the parity bit enable will take effect only after the current byte transmission is completed. | 0 |
| 8 | PEIE | RW | Parity check interrupt enable bit. This bit indicates that parity check error interrupts are allowed. | 0 |
| 7 | TXEIE | RW | TXE interrupt enable. This bit indicates that a TXE interrupt is allowed to be generated. | 0 |
| 6 | TCIE | RW | Transmit completion interrupt enable. This bit indicates that the transmit completion interrupt is allowed to be generated. | 0 |
| 5 | RXNEIE | RW | RXNE interrupt enable. This bit indicates that a RXNE interrupt is allowed to be generated. | 0 |
| 4 | IDLEIE | RW | IDLE interrupt enable. This bit allows IDLE interrupt to be generated. | 0 |
| 3 | TE | RW | Transmit enable. Setting this bit will enable the transmitter. | 0 |
| 2 | RE | RW | Receive enable. Setting this bit enables the receiver, which starts detecting the start bit on the RX pin. | 0 |
| 1 | RWU | RW | Receiver wakeup. This bit determines whether to place the USART in silent mode.<br>1: The receiver is in silent mode.<br>0: The receiver is in normal operation mode.<br>*Note 1: Before setting the RWU bit, the USART needs to receive a data byte first, otherwise it cannot be woken up by bus idle in silent mode.*<br>*Note 2: When configured as address mark wake-up, the RWU bit cannot be modified by software when RXNE is set.* | 0 |

| | | | | |
|---|---|---|---|---|
| 0 | SBK | RW | Send break bit.<br>Set this bit to send break character. It is reset by hardware on the stop bit of the break frame.<br>1: Send; 0: Do not send. | 0 |

### 14.10.5 USART Control Register 2 (USARTx_CTLR2) (x=1/2/3/4)

Offset address: 0x10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | LINEN | STOP | | CLKEN | CPOL | CPHA | LBCL | Reserved | LBDIE | LBDL | Reserved | ADD[3:0] | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved | 0 |
| 14 | LINEN | RW | The LIN mode enable bit, set to enable LIN mode. In LIN mode, the SBK bit can be used to send LIN sync break symbols and to detect LIN sync break characters. | 0 |
| [13:12] | STOP[1:0] | RW | STOP bits.<br>These two bits are used for programming the stop bits.<br>00: 1 Stop bit<br>01: 0.5 Stop bit<br>10: 2 Stop bits<br>11: 1.5 Stop bit | 0 |
| 11 | CLKEN | RW | Clock enable.<br>This bit allows the user to enable the CK pin.<br>1: Enable<br>0: Disable | 0 |
| 10 | CPOL | RW | Clock polarity setting bit. In synchronous mode, this bit can be used to select the polarity of the clock output on the SLCK pin, and in conjunction with CPHA to generate the desired clock/data sampling relationship.<br>1: Hold high on the CK pin when the bus is idle;<br>0: Hold low on the CK pin when the bus is idle.<br>*Note: This bit cannot be modified after enabling the transmit.* | 0 |
| 9 | CPHA | RW | Clock phase setting bit. In synchronous mode, this bit can be used to select the phase of the clock output on the SLCK pin, and in conjunction with the CPOL bit to produce the desired clock/data sampling relationship.<br>1: Data capture at the second edge of the clock;<br>0: Data capture at the first edge of the clock.<br>*Note: This bit cannot be modified when transmit is enabled.* | 0 |
| 8 | LBCL | RW | Last clock pulse control bit.<br>In synchronous mode, use this bit to control whether the clock pulse corresponding to that last data byte sent is output on the CK pin;<br>1: The clock pulse for the last bit of data is not | 0 |

| | | | | |
|---|---|---|---|---|
| | | | output from CK;<br>0: The clock pulse for the last bit of data will be output from CK.<br>*Note: This bit cannot be modified after enabling transmit.* | |
| 7 | Reserved | RW | Reserved | 0 |
| 6 | LBDIE | RW | LIN Break detection interrupt enable, this position bit enables interrupts caused by LBD. | 0 |
| 5 | LBDL | RW | LIN Break detection length, this bit is used to select whether the disconnect detection is 11 bits or 10 bits.<br>1: 11-bit disconnector detection.<br>0: 10-bit disconnector detection. | 0 |
| 4 | Reserved | RW | Reserved | 0 |
| [3:0] | ADD[3:0] | RW | Address field to set the USART node address of this device. Used in silent mode under multi-processor communication to wake up a USART device using the address token. | 0 |

### 14.10.6 USART Control Register 3 (USARTx_CTLR3) (x=1/2/3/4)

Offset address: 0x14

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | CTSIE | CTSE | RTSE | DMAT | DMAR | SCEN | NACK | HDSEL | IRLP | IREN | EIE |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:11] | Reserved | RO | Reserved. | 0 |
| 10 | CTSIE | RW | CTSIE interrupt enable bit, when set this bit will generate an interrupt when CTS is set. | 0 |
| 9 | CTSE | RW | CTS enable bit, setting this bit will enable CTS flow control. | 0 |
| 8 | RTSE | RW | RTS enable bit, setting this bit will enable RTS flow control. | 0 |
| 7 | DMAT | RW | DMA transmit enable bit. This position 1 uses DMA when transmitting. | 0 |
| 6 | DMAR | RW | DMA receive enable bit. This position 1 uses DMA when receiving. | 0 |
| 5 | SCEN | RW | Smart card mode enable bit. Setting 1 enables smart card mode. | 0 |
| 4 | NACK | RW | Smart card NACK enable bit, set this bit to transmit a NACK in the event of a checksum error. | 0 |
| 3 | HDSEL | RW | Half duplex mode select bit, set this bit to select half duplex mode. | 0 |
| 2 | IRLP | RW | Infrared low-power select bit, set this bit to enable low-power mode when infrared is selected. | 0 |
| 1 | IREN | RW | Infrared enable bit, set this bit to enable infrared mode. | 0 |
| 0 | EIE | RW | Error enable interrupt bit, setting this bit generates an interrupt if FE, ORE or NE is set | 0 |

| | | | | |
|---|---|---|---|---|
| | | | provided that DMAR is set. | |

### 14.10.7 USART Guard Time and Prescaler Register (USARTx_GPR) (x=1/2/3/4)

Offset address: 0x18

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| GT[7:0] | | | | | | | | PSC[7:0] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0 |
| [15:8] | GT[7:0] | RW | Protection time value field. This field specifies the protection time in baud rate clock units. In smart card mode, the transmit complete flag is set only when the protection time has elapsed. | 0 |
| [7:0] | PSC[7:0] | RW | Prescaler value field.<br>In IR low power mode, the source clock is divided by this value (All 8 bits valid) and a value of 0 indicates a hold;<br>In IR normal mode, this bit can only be set to 1;<br>In smart card mode, the source clock is divided by twice this value (Valid for the lower 5 bits) to clock the smart card, a value of 0 indicates retention. | 0 |

# Chapter 15 Inter-integrated Circuit (I2C) interface

The internal integrated circuit bus (I2C or IIC) is widely used for communication between microcontrollers and sensors and other off-chip modules. It inherently supports multi-master and multi-slave modes and can communicate at 100KHz (Standard) and 400KHz (Fast) using just two wires (SDA and SCL).

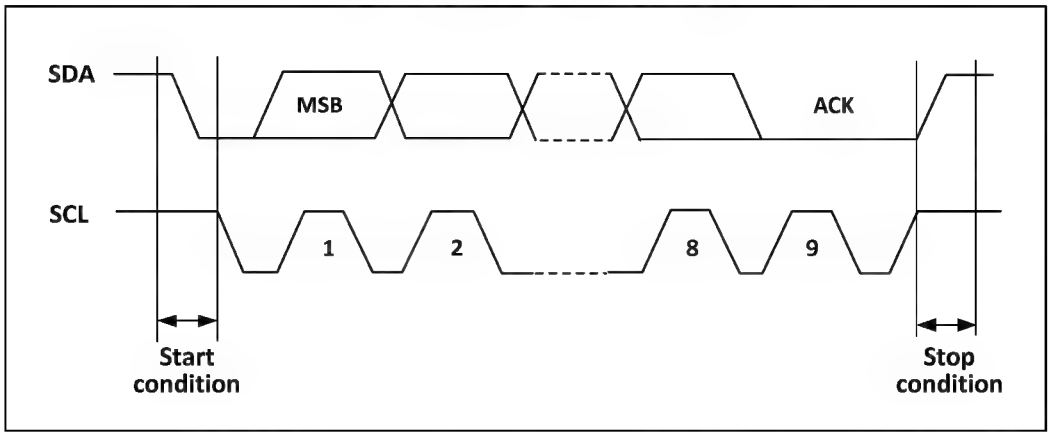
## 15.1 Main Features

- Support master and slave modes, support multiple masters and slave.
- Support two speed modes: 100KHz and 400KHz, compatible with the I2C 2-wire serial bus specification.
- Support 7-bit or 10-bit addresses.
- Slave devices support dual 7-bit addresses.
- Support bus broadcast.
- Support PEC.
- Support bus arbitration, error detection, PEC checksum, extended clock.

## 15.2 Overview

I2C is a half-duplex bus that can only operate in one of the following four modes at the same time: master device transmit mode, master device receive mode, slave device transmit mode and slave device receive mode. the I2C module works in slave mode by default and automatically switches to master mode when a start condition is generated and to slave mode when arbitration is lost or a stop signal is generated. the I2C module supports multi-master functionality. When working in master mode, the I2C module actively emits data and addresses. Both data and address are transmitted in 8-bit units, with the high bit before and the low bit after. After the start event is a one-byte (In 7-bit address mode) or two-byte (In 10-bit address mode) address, and for every 8-bit data or address sent by the host, the slave needs to reply with an answer ACK, which pulls the SDA bus low, as shown in Figure 13-1.

Figure 15-1 I2C Timing Diagram




## 15.3 Master Mode

In master mode, the I2C module dominates the data transfer and outputs the clock signal, and the data transfer

starts with a start event and ends with an end event. The steps to use master mode communication are:

1) Setting the correct clock in control register 2 (R16_I2C1_CTLR2) and clock control register (R16_I2C1_CKCFGR).
2) Setting the PE bit in the control register (R16_I2C1_CTLR1) to start the peripheral.
3) Set the START bit in the control register (R16_I2C1_CTLR1) to generate the start event.

After setting the START bit, the I2C module will automatically switch to the main mode, the MSL bit will be set and the start event will be generated. After the start event is generated, the SB bit will be set and if the ITEVTEN bit (In R16_I2C1_CTLR2) is set, an interrupt will be generated. The status register 1 (R16_I2C1_STAR1) should be read at this time and the SB bit will be cleared automatically after writing from the address to the data register.

4) If the 10-bit address mode is used, then the write data register sends the header sequence (The header sequence is 11110xx0b, where the xx bits are the top two bits of the 10-bit address).

After sending the header sequence, the ADD10 bit of the status register will be set, and if the ITEVTEN bit has been set, an interrupt will be generated, at this time the R16_I2C1_STAR1 register should be read and the ADD10 bit cleared after writing the second address byte to the data register.

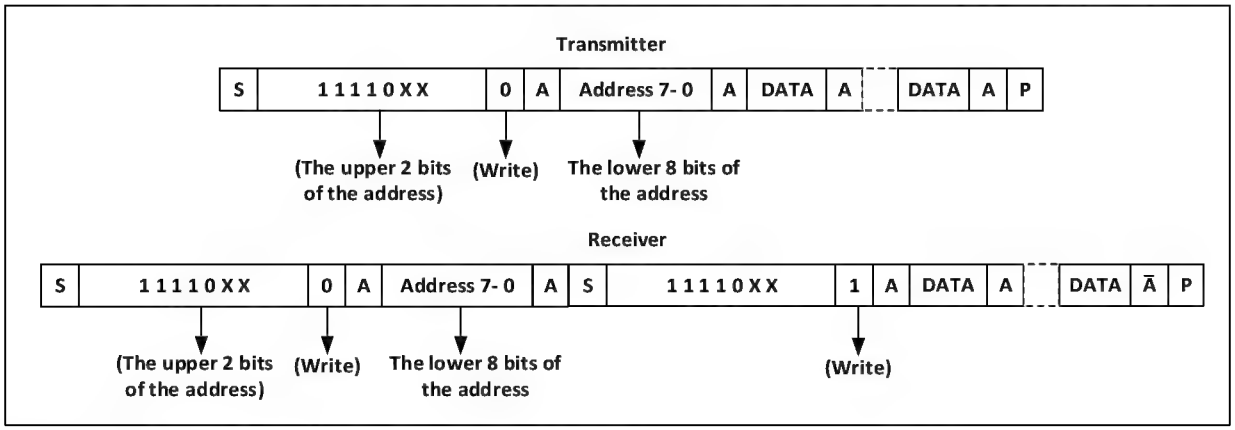
Then write the data register to send the second address byte, after sending the second address byte, the ADDR bit of the status register will be set, if the ITEVTEN bit is already set, an interrupt will be generated, at this time the R16_I2C1_STAR1 register should be read and then read the R16_I2C1_STAR2 register once to clear the ADDR bit;

If the 7-bit address mode, then write data register to send address byte, after sending address byte, ADDR bit of status register will be set, if ITEVTEN bit has been set, then interrupt will be generated, at this time, R16_I2C1_STAR1 register should be read and then R16_I2C1_STAR2 register should be read once to clear ADDR bit;

In 7-bit address mode, the first byte sent is the address byte, the first 7 bits represent the address of the target slave device, the 8th bit determines the direction of the subsequent message, 0 means the master device writes data to the slave device, 1 means the master device reads information to the slave device.

In 10-bit address mode, as shown in Figure 13-3, in the send address phase, the first byte is 11110xx0, xx is the highest 2 bits of the 10-bit address, and the second byte is the lower 8 bits of the 10-bit address. If subsequently enter the master device transmit mode, continue to send data; if subsequently ready to enter the master device receive mode, you need to re-send a start condition, follow to send a byte as 11110xx1, and then enter the master device receive mode.

Figure 15-2 Schematic diagram of master transmitting and receiving data at 10-bit address



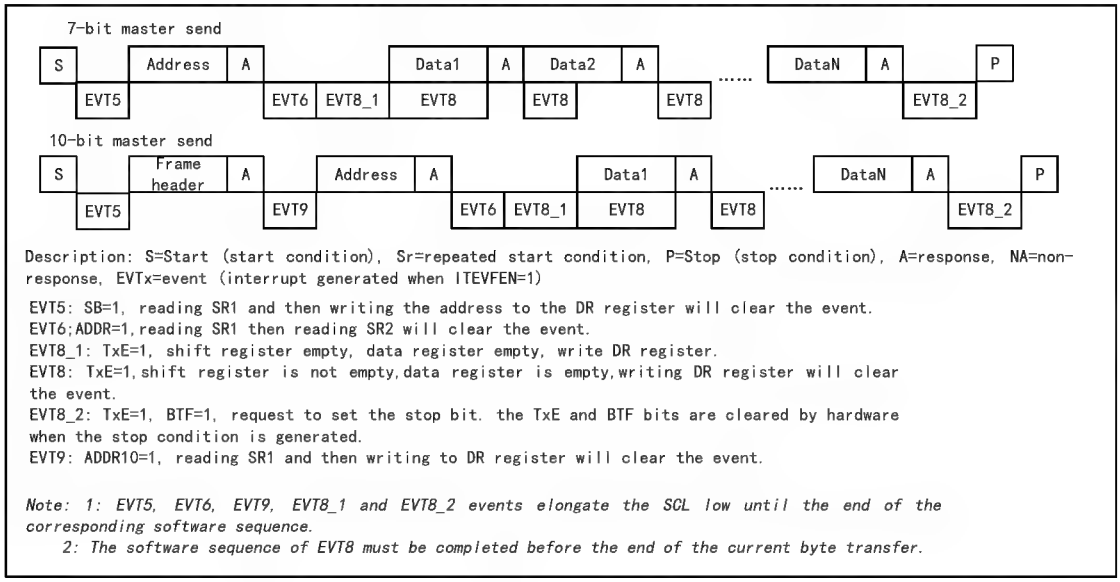

**Master transmit mode:**

The master device's internal shift register sends data from the data register to the SDA line. When the master

device receives an ACK, TxE in status register 1 (R16_I2C1_STAR1) is set, and an interrupt is also generated if ITEVTEN and ITBUFEN are set. Writing data to the data register will clear the TxE bit.

If the TxE bit is set and no new data was written to the data register before the last data was sent, then the BTF bit will be set and SCL will remain low until it is cleared, and writing data to the data register after reading R16_I2C1_STAR1 will clear the BTF bit.

Figure 15-3 Master transmitter transmission sequence diagram




Description: S=Start (start condition), Sr=repeated start condition, P=Stop (stop condition), A=response, NA=non-response, EVTx=event (interrupt generated when ITEVFEN=1)

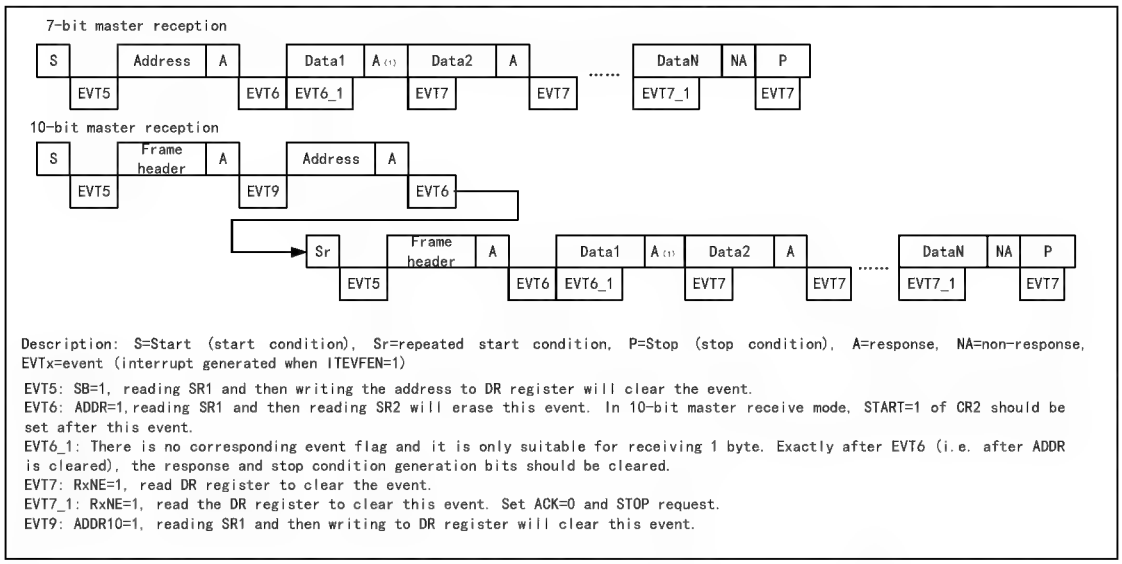
EVT5: SB=1, reading SR1 and then writing the address to the DR register will clear the event.
EVT6;ADDR=1,reading SR1 then reading SR2 will clear the event.
EVT8_1: TxE=1, shift register empty, data register empty, write DR register.
EVT8: TxE=1,shift register is not empty,data register is empty,writing DR register will clear the event.
EVT8_2: TxE=1, BTF=1, request to set the stop bit. the TxE and BTF bits are cleared by hardware when the stop condition is generated.
EVT9: ADDR10=1, reading SR1 and then writing to DR register will clear the event.

*Note: 1: EVT5, EVT6, EVT9, EVT8_1 and EVT8_2 events elongate the SCL low until the end of the corresponding software sequence.*
*2: The software sequence of EVT8 must be completed before the end of the current byte transfer.*

Master receive mode:

The I2C module will receive data from the SDA line and write it into the data register via a shift register. After each byte, if the ACK bit is set, then the I2C module will send an answer low, and the RxNE bit will be set, and an interrupt will be generated if ITEVTEN and ITBUFEN are set. If RxNE is set and the original data is not read before the new data is received, the BTF bit will be set and SCL will remain low until the BTF is cleared, and reading R16_I2C1_STAR1 and then reading the data register will clear the BTF bit.

Figure 15-4 Receiver transmission sequence diagram




Description: S=Start (start condition), Sr=repeated start condition, P=Stop (stop condition), A=response, NA=non-response, EVTx=event (interrupt generated when ITEVFEN=1)

EVT5: SB=1, reading SR1 and then writing the address to DR register will clear the event.
EVT6: ADDR=1,reading SR1 and then reading SR2 will erase this event. In 10-bit master receive mode, START=1 of CR2 should be set after this event.
EVT6_1: There is no corresponding event flag and it is only suitable for receiving 1 byte. Exactly after EVT6 (i.e. after ADDR is cleared), the response and stop condition generation bits should be cleared.
EVT7: RxNE=1, read DR register to clear the event.
EVT7_1: RxNE=1, read the DR register to clear this event. Set ACK=0 and STOP request.
EVT9: ADDR10=1, reading SR1 and then writing to DR register will clear this event.

The master device will actively send an end event, i.e. set the STOP bit, when it finishes sending data. In receive

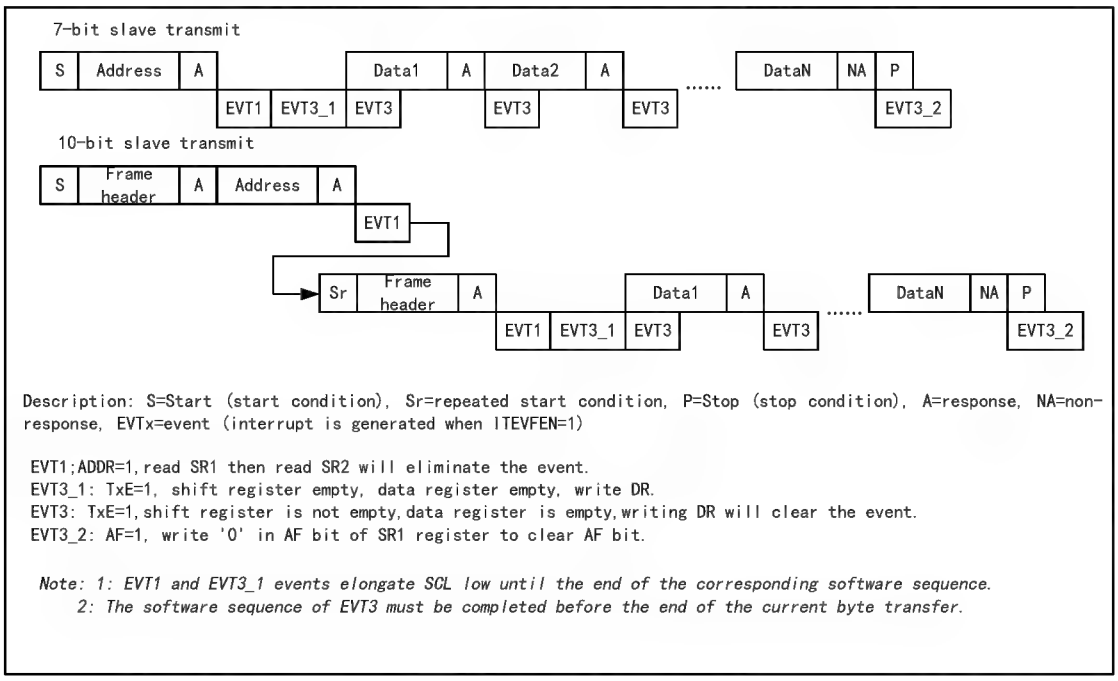
mode, the master device needs to NAK at the answer position of the last data bit. note that after generating NAK, the I2C module will switch to slave mode.

## 15.4 Slave Mode

When in slave mode, the I2C module recognizes its own address and the broadcast call address. The software can control whether the recognition of the broadcast call address is enabled or disabled. Once a start event is detected, the I2C module compares the SDA data through the shift register with its own address (Number of bits depends on ENDUAL and ADDMODE) or the broadcast address (When ENGC is set), if there is a mismatch it will be ignored until a new start event is generated. If it matches the header sequence, an ACK signal is generated and the address of the second byte is waited for; if the address of the second byte also matches or the full segment address matches in the case of a 7- bit address, then:
First an ACK answer is generated;
The ADDR bit is set, and if the ITEVTEN bit is already set, then a corresponding interrupt is also generated;
if the dual address mode is used (ENDUAL bit is set), the DUALF bit also needs to be read to determine which address the host is evoking.
The slave mode is receive mode by default. In case the last bit of the received header sequence is 1, or the last bit of the 7-bit address is 1 (Depending on whether the header sequence is received for the first time or a normal 7-bit address), the I2C module will go to transmitter mode and the TRA bit will indicate whether it is currently receiver or transmitter mode.

**Slave transmit mode:**
After clearing the ADDR bit, the I2C module sends bytes from the data register to the SDA line via a shift register. After an answer ACK is received, the TxE bit is set and an interrupt is generated if ITEVTEN and ITBUFEN are set. If TxE is set but no new data is written to the data register before the end of the next data send, the BTF bit will be set. SCL will remain low until the BTF is cleared. Reading status register 1 (R16_I2C1_STAR1) and then writing data to the data register will clear the BTF bit.

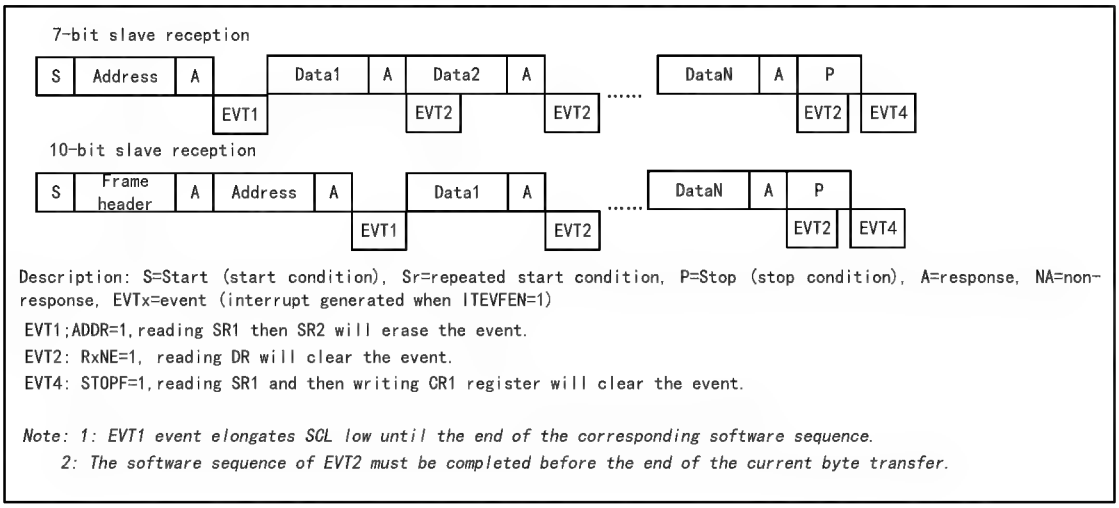
Figure 15-5 Slave transmitter transmission sequence diagram




**Slave receive mode:**

After ADDR is cleared, the I2C module stores the data on SDA into the data register via the shift register. After each byte is received, the I2C module sets an ACK bit and sets the RxNE bit, and generates an interrupt if ITEVTEN and ITBUFEN are set. If RxNE is set and the old data is not read before the new data is received, then BTF is set. SCL will remain low until the BTF bit is cleared. Reading status register 1 (R16_I2C1_STAR1) and reading the data in the data register will clear the BTF bit.

Figure 15-6 Receiver transmission sequence diagram




The master device will generate a stop condition after the last data byte is transferred. When the I2C module detects a stop event, it will set the STOPF bit, and if the ITEVFEN bit is set, it will also generate an interrupt. The user needs to read the status register (R16_I2C1_STAR1) and then write the control register (e.g. reset control word SWRST) to clear it.

## 15.5 Error Conditions

### 15.5.1 Bus Error (BERR)

A bus error will be generated when the I2C module detects an external start or stop event during address or data transfer. When a bus error is generated, the BERR bit is set and an interrupt is generated if ITERREN is set. In slave mode, the data is discarded and the hardware releases the bus. If it is a start signal, the hardware assumes it is a restart signal and starts waiting for an address or stop signal; if it is a stop signal, it operates ahead of normal stop conditions. In master mode, the hardware does not release the bus while not affecting the current transfer, and it is up to the user code to decide whether to abort the transfer.

### 15.5.2 Acknowledge Failure (AF)

An answer error will be generated when the I2C module detects a byte and then no answer. When an answer error is generated: AF will be set and an interrupt will be generated if ITERREN is set; when an AF error is encountered, the hardware must release the bus if the I2C module is working in slave mode and the software must generate a stop event if it is in master mode.

### 15.5.3 Arbitration Lost (ARLO)

An arbitration lost error is generated when the I2C module detects an arbitration lost. When an arbitration loss error is generated: the ARLO bit is set and an interrupt is generated if ITERREN is set; the I2C module switches to slave mode and no longer responds to transfers initiated against its slave address unless a new start event is initiated by the host; the hardware releases the bus.

### 15.5.4 Overrun/Underrun Error (OVR)

- **Overrun error**

In Slave mode, if the clock extension is disabled and the I2C module is receiving data, an overrun error will occur if a byte of data has been received but the last received data has not been read out. When an overrun error occurs, the last received byte will be discarded and the sender should retransmit the last sent byte.

- **Underrun error**

In Slave mode, if the clock is forbidden to extend and the I2C module is sending data, an underrun error will occur if new data has not been written to the data register before the next byte of the clock comes. In case of an underrun error, the data in the previous data register will be sent twice, and if an underrun error occurs, then the receiver should discard the data received repeatedly. In order not to generate an underrun error, the I2C module should write the data to the data register before the first rising edge of the next byte.

## 15.6 Clock Extension

If clock extension is disabled, then there is a possibility of overrun/underrun errors. However, if clock extension is enabled:

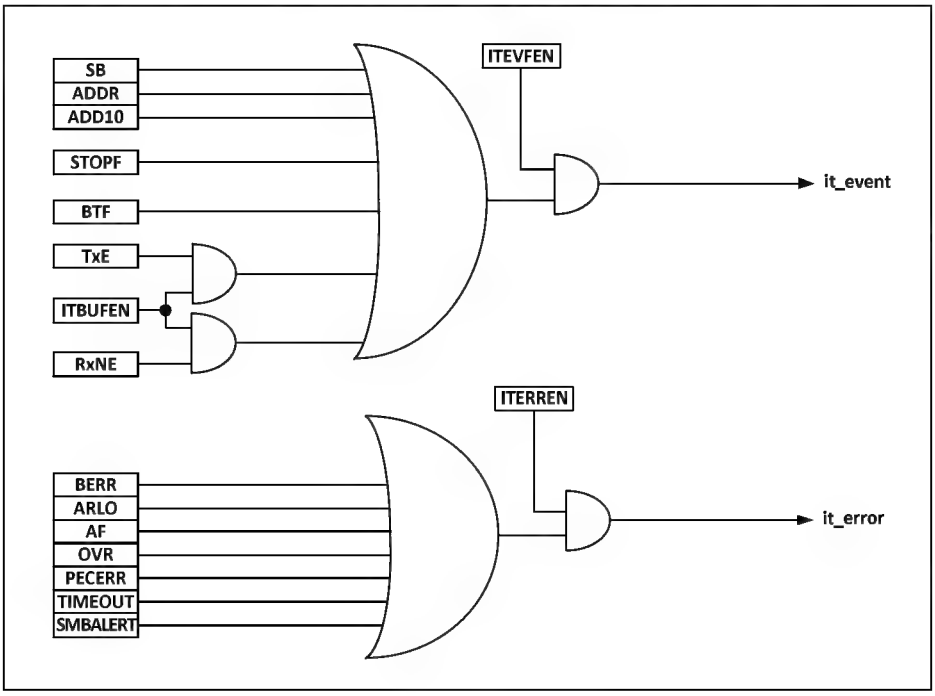
- In transmit mode, if TxE is set and BTF is set, SCL will always be low, always waiting for the user to read the status register and write the data to be sent to the data register.
- In receive mode, if RxNE is set and BTF is set, SCL will remain low after data is received until the user reads the status register and reads the data register.

It can be seen that enabling clock extension can avoid overrun/underrun errors.

## 15.7 Interrupts

Each I2C module has 2 interrupt vectors, event interrupts and error interrupts. Both interrupts support the interrupt sources in Figure 15-7.

Figure 15-7 I2C Interrupt Request




## 15.8 Packet Error Checking

Packet Error Checksum (PEC) is an additional CRC8 checksum step to provide transmission reliability, calculated for each bit of serial data using the following polynomial.

$$C=X^8+X^2+X+1$$

The PEC calculation is activated by the ENPEC bit in the control register and is performed on all information bytes, including address and read/write bits. In transmitting, enabling PEC adds a byte of CRC8 calculation result after the last byte of data; while in receiving mode, in the last byte is considered as CRC8 check result, and if it does not match with the internal calculation result, it will reply a NAK, and in case of the main receiver, regardless of the correct check result.

## 15.9 Register Description

Table 15-1 I2C-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_I2C1_CTLR1 | 0x40005400 | I2C control register 1 | 0x0000 |
| R16_I2C1_CTLR2 | 0x40005404 | I2C control register 2 | 0x0000 |
| R16_I2C1_OADDR1 | 0x40005408 | I2C address register 1 | 0x0000 |
| R16_I2C1_OADDR2 | 0x4000540C | I2C address register 2 | 0x0000 |

| | | | |
|---|---|---|---|
| R16_I2C1_DATAR | 0x40005410 | I2C data register | 0x0000 |
| R16_I2C1_STAR1 | 0x40005414 | I2C status register 1 | 0x0000 |
| R16_I2C1_STAR2 | 0x40005418 | I2C status register 2 | 0x0000 |
| R16_I2C1_CKCFGR | 0x4000541C | I2C clock register | 0x0000 |

### 15.9.1 I2C Control Register 1 (I2C1_CTLR1)

Offset address: 0x00

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SWRST | Reserved | | PEC | POS | ACK | STOP | START | NOSTRETCH | ENGC | ENPEC | ENARP | Reserved | | | PE |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | SWRST | RW | Software reset, setting this bit by user code will reset the I2C peripheral. Make sure the pins of the I2C bus are released and the bus is idle before the reset.<br>*Note: This bit resets the I2C module when no stop condition is detected on the bus but the busy bit is 1.* | 0 |
| [14:13] | Reserved | RO | Reserved | 0 |
| 12 | PEC | RW | Packet error checking bit, set this bit to enable packet error detection. The user code can set or clear this bit; the hardware clears this bit when the PEC is transmitted, when a start or end signal is generated, or when the PE bit is cleared to 0.<br>1: With PEC.<br>0: Without PEC.<br>*Note: The PEC is invalidated when arbitration is lost.* | 0 |
| 11 | POS | RW | ACK and PEC position setting bits, which can be set or cleared by user code and can be cleared by hardware after the PE has been cleared.<br>1: ACK bit controls the ACK or NAK of the next byte received in the shift register. The next byte received in the PEC shift register is the PEC.<br>0: ACK bit controls the ACK or NAK of the byte currently being accepted in the shift register. the PEC bit indicates that the byte in the shift register before the current bit is PEC.<br>*Note: The POS bit is used in 2-byte data reception as follows: it must be configured before reception. In order to NACK the 2nd byte, the ACK bit must be cleared immediately after clearing the ADDR bit; in order to detect the PEC of the second byte, the PEC bit must be set after the ADDR event and after configuring the POS bit.* | 0 |
| 10 | ACK | RW | Acknowledge enable, This bit is set and cleared by software and cleared by hardware when PE=0.<br>1: Acknowledge returned after a byte is received.<br>0: No acknowledge returned. | 0 |
| 9 | STOP | RW | Stop generation bit. This bit is set and cleared by software, cleared by hardware when a Stop | 0 |

| | | | | |
|---|---|---|---|---|
| | | | condition is detected, set by hardware when a timeout error is detected.<br>In Master mode:<br>1: Stop generation after the current byte transfer or after the current Start condition is sent.<br>0: No Stop generation.<br>In Slave mode:<br>1: Release the SCL and SDA lines after the current byte transfer.<br>0: No Stop generation. | |
| 8 | START | RW | Start generation. This bit is set and cleared by software and cleared by hardware when start is sent or PE=0.<br>In Master mode:<br>1：Repeated start generation<br>0：No Start generation<br>In Slave mode:<br>1：Start generation when the bus is free<br>0：No Start generation | 0 |
| 7 | NOSTRETCH | RW | Clock stretching disable bit. This bit is used to disable clock stretching in slave mode when ADDR or BTF flag is set, until it is reset by software.<br>1：Clock stretching disabled.<br>0：Clock stretching enabled. | 0 |
| 6 | ENGC | RW | General call enable bit. Set this bit to enable broadcast call and answer broadcast address 00h. | 0 |
| 5 | ENPEC | RW | PEC enable bit, set this bit to enable PEC calculation. | 0 |
| 4 | ENARP | RW | ARP enable bit, set this bit to enable ARP. | 0 |
| [3:1] | Reserved | RO | Reserved | 0 |
| 0 | PE | RW | I2C peripheral enable bit.<br>1: Enable the I2C module.<br>0: Disable the I2C module. | 0 |

### 15.9.2 I2C Control Register 2 (I2C1_CTLR2)

Offset address: 0x04

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | LAST | DMA EN | ITBU FEN | ITEV TEN | ITER REN | Reserved | | FREQ[5:0] | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:13] | Reserved | RO | Reserved. | 0 |
| 12 | LAST | RW | DMA last transfer bit.<br>1: Next DMA EOT is the last transfer.<br>0: Next DMA EOT is not the last transfer.<br>*Note: This bit is used in master receiver mode to permit the generation of a NACK on the last received data.* | 0 |
| 11 | DMAEN | RW | DMA requests enable bit. Set this bit to allow DMA request when TxE or RxEN is set. | 0 |
| 10 | ITBUFEN | RW | Buffer interrupt enable bit.<br>1: When TxE or RxEN is set, event interrupt is | 0 |

| | | | | |
|---|---|---|---|---|
| | | | generated.<br>0: When TxE or RxEN is set, no interrupt is generated. | |
| 9 | ITEVTEN | RW | Event interrupt enable bit. Set this bit to enable event interrupt. This interrupt will be generated under the following conditions.<br>SB=1 (Master mode).<br>ADDR=1 (Master-slave mode).<br>ADDR10 = 1 (Master mode).<br>STOPF=1 (Slave mode).<br>BTF = 1, but no TxE or RxEN events.<br>TxE event to 1 if ITBUFEN = 1.<br>RxNE event to 1if ITBUFEN = 1. | 0 |
| 8 | ITERREN | RW | Error interrupt enable bit. Set to allow error interrupts.<br>The interrupt will be generated under the following conditions.<br>BERR=1; ARLO=1; AF=1; OVR=1; PECERR=1.<br>TIMEOUT=1; SMBAlert=1. | 0 |
| [7:6] | Reserved | RO | Reserved | 0 |
| [5:0] | FREQ[5:0] | RW | The I2C module clock frequency field, which must be entered at the correct clock frequency to produce the correct timing, allows a range between 4-60 MHz. It must be set between 000100b and 111100b in MHz. | 0 |

### 15.9.3 I2C Address Register 1 (I2C1_OADDR1)

Offset address: 0x08

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDMODE | Reserved | | | | | ADD[9:8] | | ADD[7:1] | | | | | | | ADD0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | ADDMODE | RW | Address mode.<br>1: 10-bit slave address (Does not respond to 7-bit addresses).<br>0: 7-bit slave address (Does not respond to 10-bit address) | 0 |
| [14:10] | Reserved | RO | Reserved | 0 |
| [9:8] | ADD[9:8] | RW | Interface address, bits 9-8 when using a 10-bit address, ignored when using a 7-bit address. | 0 |
| [7:1] | ADD[7:1] | RW | Interface address, bits 7-1. | 0 |
| 0 | ADD0 | RW | Interface address, bit 0 when using a 10-bit address, ignored when using a 7-bit address. | 0 |

### 15.9.4 I2C Address Register 2 (I2C1_OADDR2)

Offset address: 0x0C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | ADD2[7:1] | | | | | | | ENDUAL |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved | 0 |
| [7:1] | ADD2[7:1] | RW | Interface address, bits 7-1 of the address in dual address mode. | 0 |
| 0 | ENDUAL | RW | Dual address mode enable bit, set this bit to allow ADD2 to be recognized as well. | 0 |

### 15.9.5 I2C Data Register (I2C1_DATAR)

Offset address: 0x10

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | DR[7:0] | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved | 0 |
| [7:0] | DR[7:0] | RW | Data register, this field is used to store the received data or to store the data used to send to the bus. | 0 |

### 15.9.6 I2C Status Register 1 (I2C1_STAR1)

Offset address: 0x14

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | PECERR | OVR | AF | ARLO | BERR | TxE | RxNE | Reserved | STOPF | ADD10 | BTF | ADDR | SB |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:13] | Reserved | RO | Reserved | 0 |
| 12 | PECERR | RW0 | The PEC error flag bit occurs on reception, and this bit can be reset by a user write of 0 or by hardware when PE goes low.<br>1: There is a PEC error and the PEC is received and NAK is returned.<br>0: No PEC error. | 0 |
| 11 | OVR | RW0 | Overrun and underrun flag bits.<br>1: There are overrun and underrun events occurring: when NOSTRETCH=1, when a new byte is received in receive mode, the content in the data register has not been read out, then the newly received byte will be lost; when in send mode, no new data is written to the data register, and the same byte will be sent twice.<br>0: No overrun or underrun events. | 0 |
| 10 | AF | RW0 | Acknowledge failure bit. Cleared by software writing 0, or by hardware when PE=0.<br>1: Acknowledge failure.<br>0: No acknowledge failure. | 0 |
| 9 | ARLO | RW0 | Arbitration loss flag bit, which can be reset by a user write of 0, or by hardware when PE goes low.<br>1: Loss of arbitration detected and the module loses control of the bus; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: Arbitration normal. | |
| 8 | BERR | RW0 | The bus error flag bit, which can be reset by a user write of 0, or by hardware when PE goes low.<br>1: Error in start or stop condition;<br>0: Normal. | 0 |
| 7 | TxE | RO | The data register is the empty flag bit, which can be cleared by writing data to the data register, either after generating a start or stop bit, or automatically by hardware when PE is 0.<br>1: The transmit data register is empty when data is sent;<br>0: Data register is non-empty. | 0 |
| 6 | RxNE | RO | Data register not empty bit. Cleared by software reading or writing the DR register or by hardware when PE=0.<br>1: Data register not empty.<br>0: Data register empty. | 0 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | STOPF | RO | Stop detection bit. Cleared by software reading the SR1 register followed by a write in the CR1 register, or by hardware when PE=0<br>1: Set by hardware when a Stop condition is detected on the bus by the slave after an acknowledge (If ACK=1).<br>0: No Stop condition detected. | 0 |
| 3 | ADD10 | RO | 10-bit header sent bit. Cleared by software reading the SR1 register followed by a write in the DR register of the second address byte, or by hardware when PE=0.<br>1: Master has sent first address byte.<br>0: No ADD10 event occurred. | 0 |
| 2 | BTF | RO | End of byte send flag bit, read or write to the data register will clear this bit after the user reads status register 1; in transmission, after initiating a start or stop event, or when PE is 0, this bit is cleared by hardware.<br>1: End of byte transmission.<br>When NOSTRETCH=0: on transmit, when a new data is sent and the data register has not yet been written with new data; on receive, when a new byte is received but the data register has not yet been read;<br>0: None. | 0 |
| 1 | ADDR | RW0 | Address sent /matched bit. This bit is cleared by software reading SR1 register followed reading SR2, or by hardware when PE=0.<br>In Master mode:<br>1：End of address transmission. For 10-bit addressing, the bit is set after the ACK of the 2nd byte. For 7-bit addressing, the bit is set after the ACK of the byte.<br>0：No end of address transmission.<br>In Slave mode:<br>1：Received address matched.<br>0：Address mismatched or not received. | 0 |

| | | | | |
|---|---|---|---|---|
| 0 | SB | RO | The start bit transmit flag bit, an operation to write the data register after reading status register 1 will clear this bit, or the hardware will clear this bit when PE is 0.<br>1: The start bit has been transmitted;<br>0: The start bit not transmitted. | 0 |

### 15.9.7 I2C Status Register 2 (I2C1_STAR2)

Offset address: 0x18

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PEC[7:0] | | | | | | | | DUALF | Reserved | | GENCALL | Reserved | TRA | BUSY | MSL |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | PEC[7:0] | RO | Packet error check field, when PEC is enabled (ENPEC is set) this field holds the value of PEC. | 0 |
| 7 | DUALF | RO | The match detection flag bit, which is cleared by the hardware when a stop or start bit is generated, or when PE=0.<br>1: The received address matches the contents of OAR2;<br>0: The received address matches the contents in OAR1. | 0 |
| [6:5] | Reserved | RO | Reserved. | 0 |
| 4 | GENCALL | RO | The broadcast call address flag bit, which is cleared by the hardware when a stop or start bit is generated, or when PE=0.<br>1: The address of the broadcast call received when ENGC=1;<br>0: Address of broadcast call not received. | 0 |
| 3 | Reserved | RO | Reserved | 0 |
| 2 | TRA | RO | The transmit/receive flag bit, which is cleared by the hardware when a stop event is detected (STOPF=1), a repeated start condition, a bus arbitration loss (ARLO=1) or when PE=0.<br>1: Data transmitted;<br>0: Data received.<br>This bit is determined according to the RW bit of the address byte. | 0 |
| 1 | BUSY | RO | Busy flag bit, this bit will be cleared when a stop bit is detected. It is still updated when the interface is disabled (PE=0).<br>1: Bus busy: low level present in SDA or SCL;<br>0: Bus idle no communication. | 0 |
| 0 | MSL | RO | Master-slave mode indicator bit, hardware sets this bit when the interface is in master mode (SB=1); hardware clears this bit when the bus detects a stop bit, when arbitration is lost, or when PE=0. | 0 |

### 15.9.8 I2C Clock Register (I2C1_CKCFGR)

Offset address: 0x1C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F/S | DUTY | Reserved | | CCR[11:0] | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | F/S | RW | Master mode selection bit.<br>1：Fm mode I2C.<br>0：Sm mode I2C | 0 |
| 14 | DUTY | RW | Duty cycle in the fast mode:<br>1: $T_{Low\ level}/T_{High\ level}$=16/9;<br>0: $T_{Low\ level}/T_{High\ level}$=2. | 0 |
| [13:12] | Reserved | RO | Reserved | 0 |
| [11:0] | CCR[11:0] | RW | The clock division factor field, which determines the frequency waveform of the SCL clock. | 0 |

# Chapter 16 Serial Peripheral Interface (SPI)

SPI supports data interaction in a 3-wire synchronous serial mode, plus a chip selector line to support hardware switching between Master and Slave modes, and supports communication on a single data line.

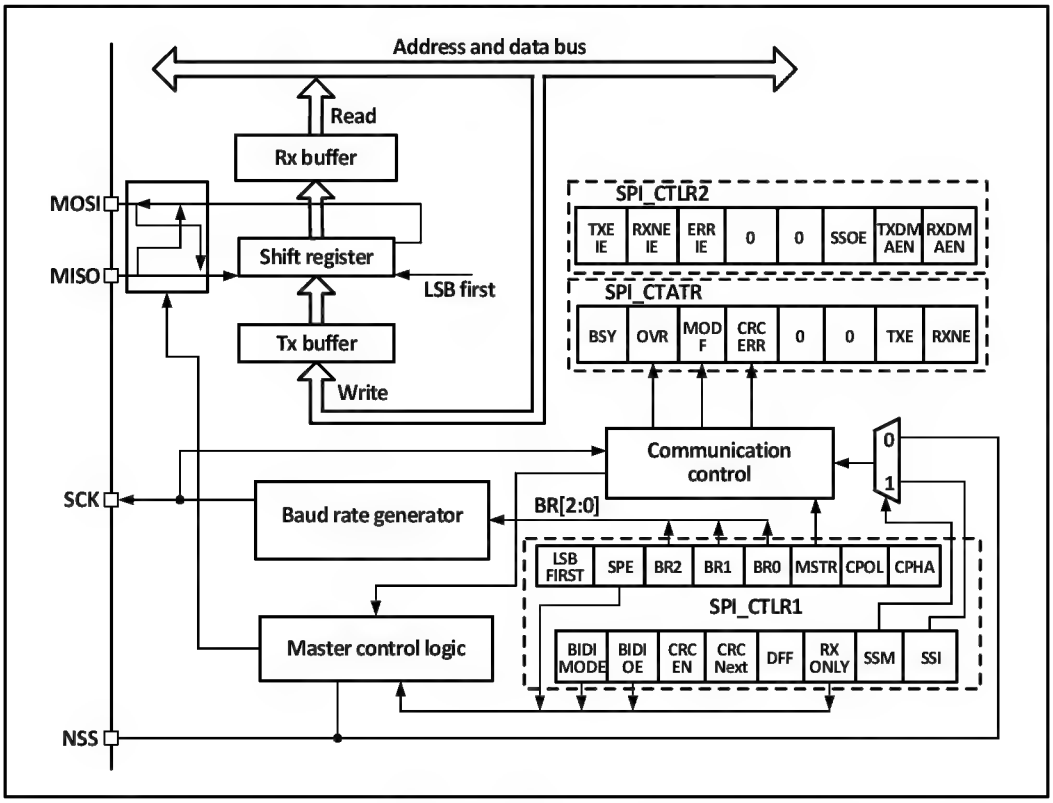
## 16.1 Main Features

- Support full-duplex synchronous serial mode
- Support single-line half-duplex mode
- Support Master mode and Slave mode, multi-slave mode
- Support 8-bit or 16-bit data structures
- Maximum clock frequency supports up to half of $F_{HCLK}$
- Data order supports MSB or LSB first
- Support hardware or software control of NSS pins
- Hardware CRC checksum support for transmitting and receiving
- Transceiver buffers support DMA transfers
- Support modification of clock phase and polarity

## 16.2 Function Description

### 16.2.1 Overview

Figure 16-1 SPI structure block diagram




As can be seen from Figure 16-1, the four main SPI-related pins are MISO, M0SI, SCK and NSS. The MISO pin

is the data input pin when the SPI module is operating in Master mode and the data output pin when it is operating in Slave mode. the MOSI pin is the data output pin when it is operating in Master mode and the data input pin when it is operating in Slave mode. the SCK is the clock pin, the clock signal is always output by the host and the slave receives the clock signal and synchronizes the data sending and receiving. the NSS pin is the chip select pin with the following usage.

1) NSS controlled by software: when SSM is set and the internal NSS signal is output high or low as determined by SSI, this case is generally used in SPI Master mode.
2) NSS is controlled by hardware: when NSS output is enabled, i.e., when SSOE is set, the NSS pin will be actively pulled low when the SPI host sends output outward, and a hardware error will be generated if the NSS pin is pulled low; if SSOE is not set, it can be used in Multi-master mode, and if it is pulled low it will be forced into Slave mode and the MSTR bit will be cleared automatically.

The operating mode of the SPI can be configured via CPHA and CPOL; CPHA is set to indicate that the module samples data on the second edge of the clock and the data is latched, CPHA is not set to indicate that the SPI module samples data on the first edge of the clock and the data is latched, and CPOL indicates whether the clock is held high or low when there is no data.

The host and device need to be set to the same SPI mode and the SPE bit needs to be cleared before configuring the SPI mode. the DEF bit determines whether the individual data length of the SP is 8 or 16 bits. the LSBFIRST controls whether the individual data word is preceded by a high bit or a low bit.

Table 16-1 SPI mode differences

| | Mode 0 | Mode 1 | Mode 2 | Mode 3 |
|---|---|---|---|---|
| CPOL | 0 | 1 | 1 | 1 |
| CPHA | 0 | 0 | 0 | 1 |

### 16.2.2 Master Mode

The serial clock is generated by SCK when the SPI module is operating in master mode. The following steps are performed to configure into master mode:

The BR[2:0] field of the configuration control register to determine the clock;

Configure the CPOL and CPHA bits to determine the SPI mode;

Configure DEF to determine the data word length;

Configure LSBFIRST to determine the frame format;

Configure the NSS pin, for example by setting the SSOE bit and letting the hardware set the NSS. it is also possible to set the SSM bit and set the SSI bit high;

To set the MSTR bit and the SPE bit, you need to make sure that the NSS is already high at this time.

When data needs to be sent just write the data to be sent to the data register. SPI will send the data from the send buffer to the shift register in parallel, when the data has reached the shift register, the TXE flag will be set, if the TXEIE has been set, then an interrupt will be generated. If the TXE flag position bit needs to fill the data register with data to maintain the complete data flow.

When the receiver receives data, when the last sample clock edge of the data word comes, the data is transferred from the shift register to the receive buffer in parallel, the RXNE bit is set, and an interrupt is generated if the RXNEIE bit was previously set. At this time, the data register should be read as soon as possible to take away the data.

Figure 16-2 SPI master mode read/write mode 0

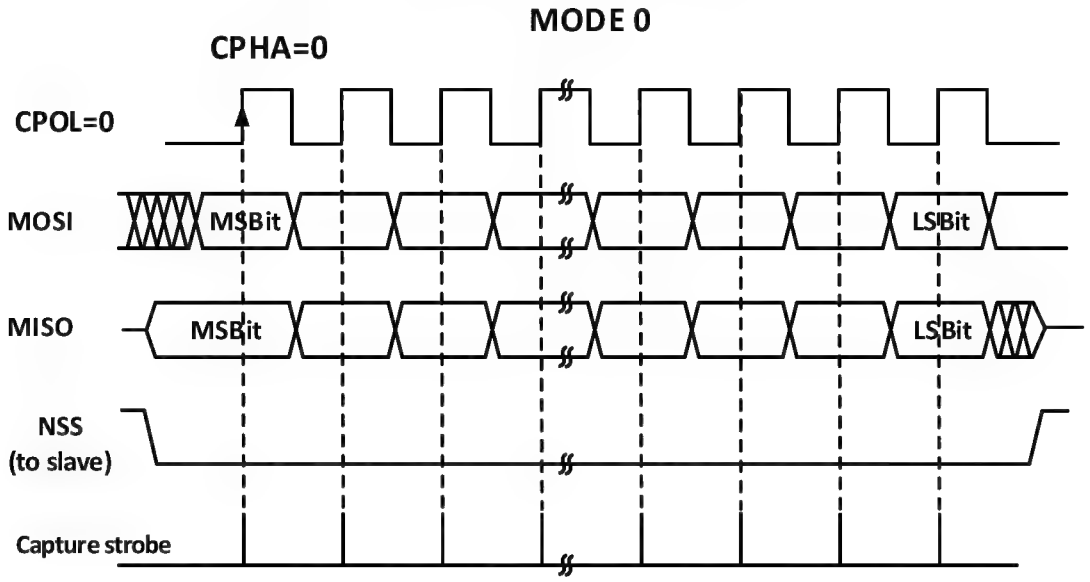



Figure 16-3 SPI master mode read/write mode 1

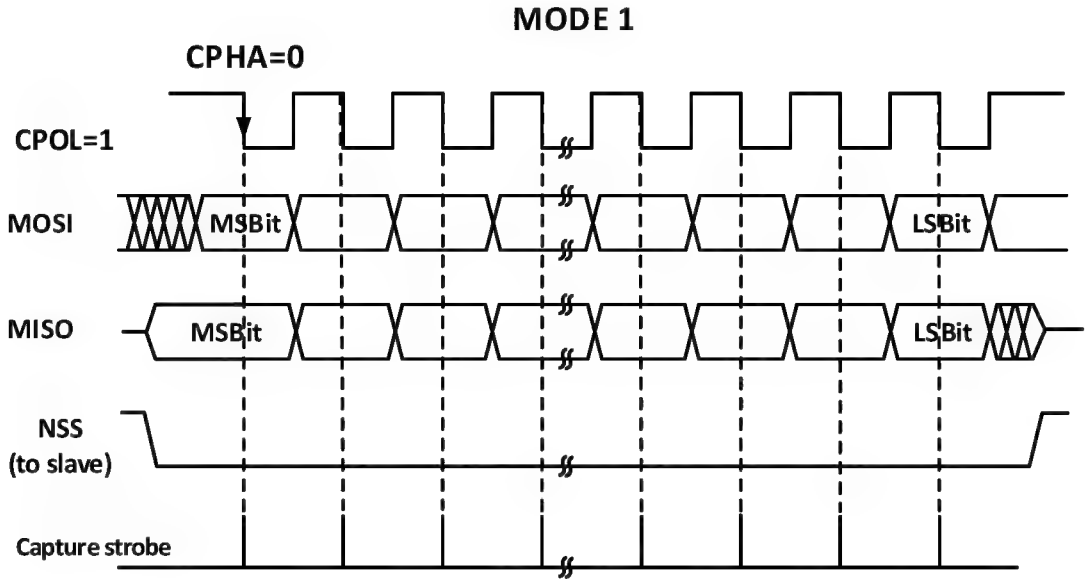


Figure 16-4 SPI master mode read/write mode 2

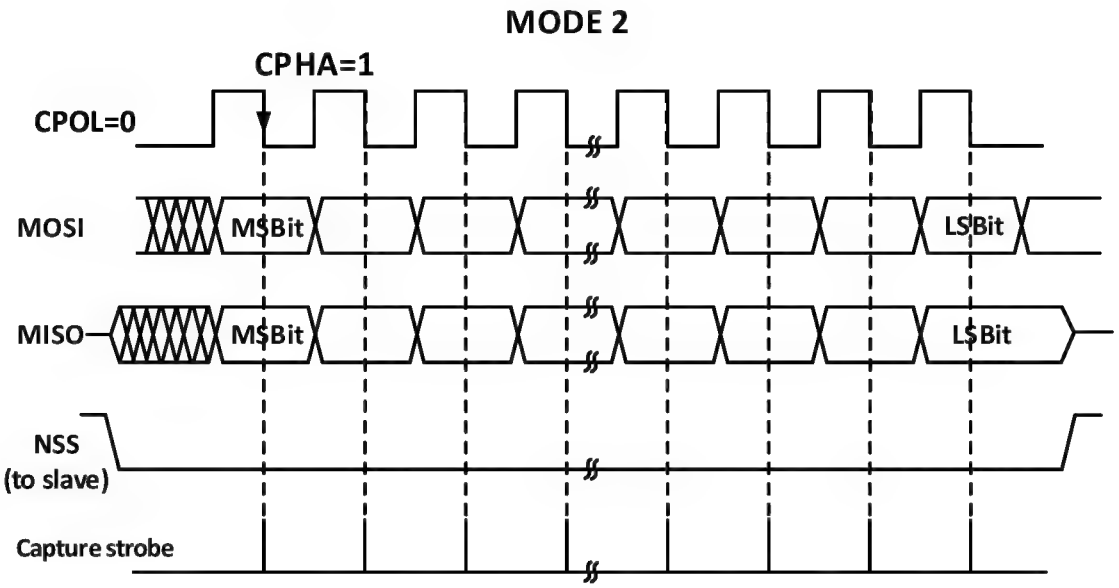
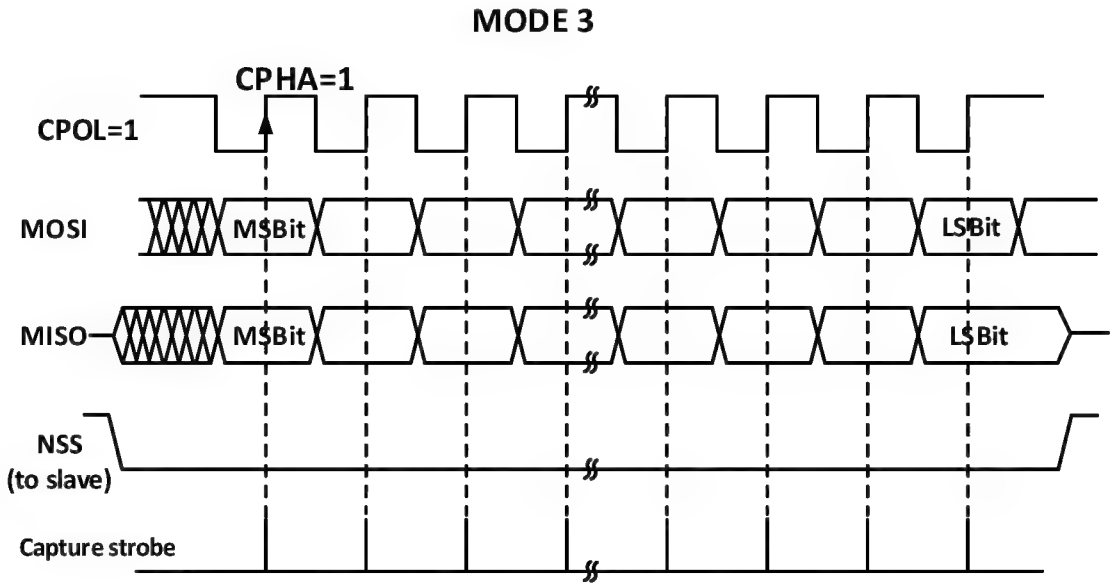



Figure 16-5 SPI master mode read/write mode 3




### 16.2.3 Slave Mode

When the SPI module is operating in slave mode, SCK is used to receive the clock from the host and its own baud rate setting is invalid. To configure into slave mode, proceed as follows.

Configure the DEF bit to set the data bit length.

Configure CPHA to match the host operating mode;

Determine the SPI operating mode based on the transmitting and receiving configurations and CPOL;

If transmitting in slave mode is required, CPOL needs to be set and configured as mode 2 or mode 3, and the host changes the configuration as required;

If only receiving in slave mode is required, only the host CPOL mode needs to be matched;

Configure LSBFIRST to match the host data frame format;

The NSS pin needs to be held low in hardware management mode, if NSS is set to software management (SSM set), then keep SSI unset.

Clear the MSTR bit and set the SPE bit to enable SPI mode.

During transmission, when the first slave receive sample edge appears in SCK, the slave starts to transmit. The process of transmitting is to move the data in the transmit buffer t the transmit shift register. When the data in the transmit buffer is moved to the shift register, the TXE flag will be set, and if the TXEIE bit was set before, then an interrupt will be generated.

During reception, after the last clock sample edge, the RXNE bit is set, the bytes received by the shift register are transferred to the receive buffer, and the read operation of the read data register can obtain the data in the receive buffer. If RXNEIE is set before RXNE is set, then an interrupt is generated.

Figure 16-6 SPI slave mode read mode 0

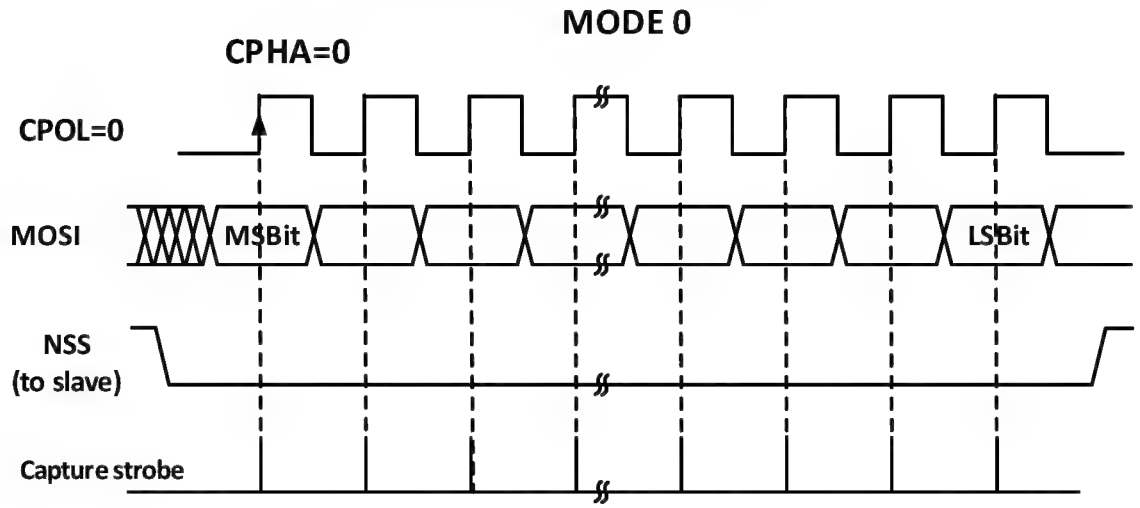



Figure 16-7 SPI slave mode read mode 1

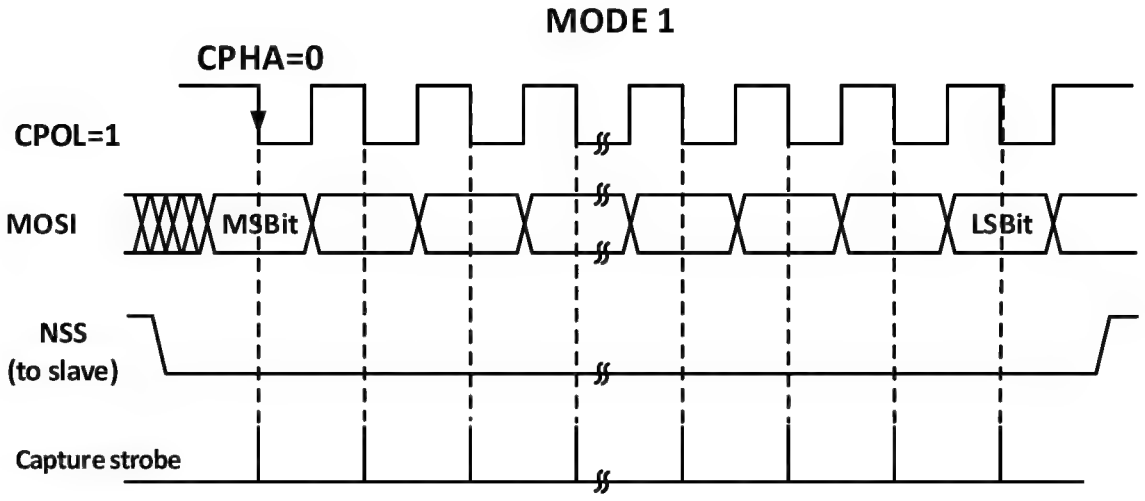


Figure 16-8 SPI slave mode read/write mode 2

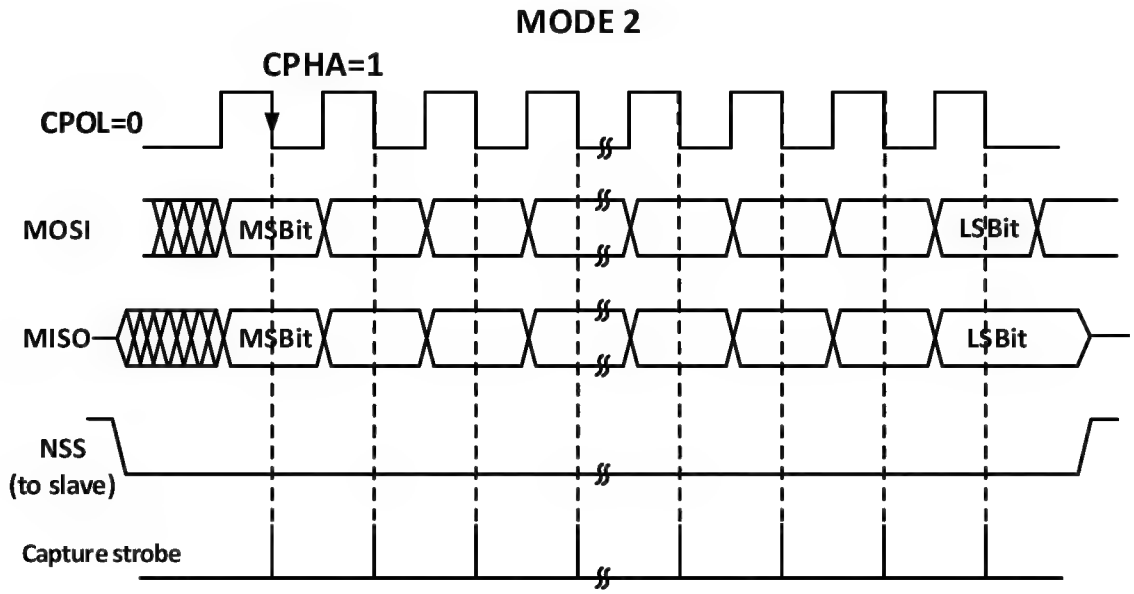
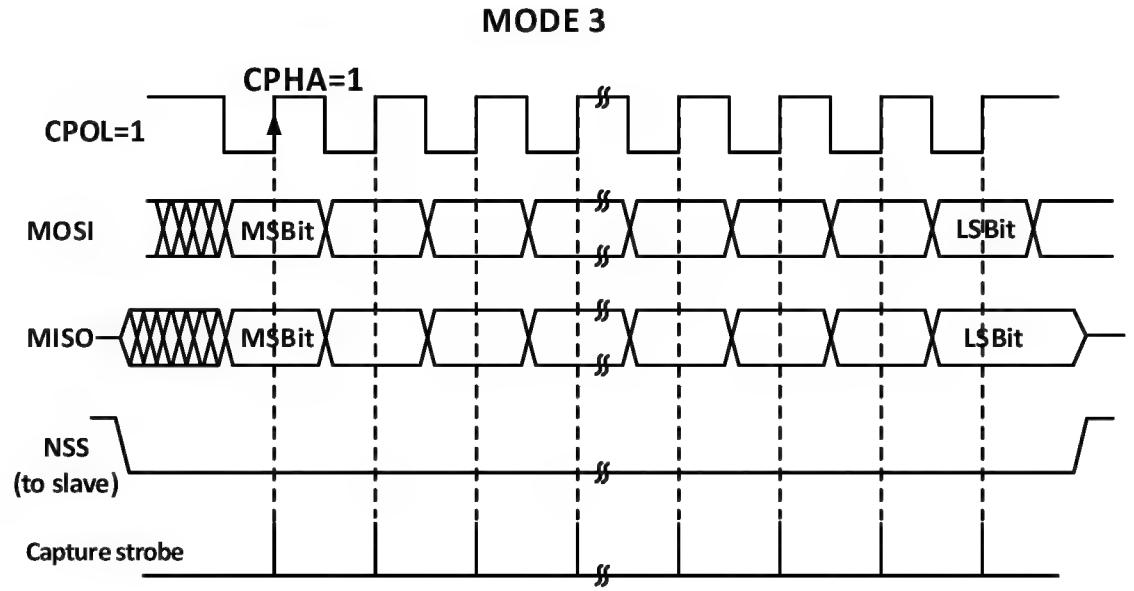



Figure 16-9 SPI slave mode read/write mode 3




### 16.2.4 Simplex Mode

The SPI interface can operate in half-duplex mode, where the master device uses the MOSI pin and the slave device uses the MISO pin for communication. When using half-duplex communication, you need to set BIDIMODE and use BIDIOE to control the transmission direction.

Setting the RXONLY bit in normal full-duplex mode sets the SPI module to receive-only simplex mode, releasing a data pin after RXONLY is set. The SPI can also be set to transmit only mode by ignoring the received data.

### 16.2.5 CRC

The SPI module uses CRC checksum to ensure the reliability of full-duplex communication, and separate CRC calculators are used for data transmitting and receiving. the polynomial for CRC calculation is determined by the

polynomial register, and different calculations are used for 8-bit data width and 16-bit data width, respectively. Setting the CRCEN bit will enable CRC checksum and at the same time will reset the CRC calculator. After the last data byte is transmitted, setting the CRCNEXT bit will transmit the TXCRCR calculator calculation after the current byte is sent, while the CRCERR bit will be set if the last received receive shift register value does not match the locally calculated RXCRCR calculation. Using the CRC checksum requires setting the polynomial calculator and setting the CRCEN bit when configuring the SPI operating mode, and setting the CRCNEXT bit on the last word or half-word to transmit the CRC and perform the receive CRC checksum. Note that the polynomial for the CRC calculation should be unified for both transmitting and receiving.

### 16.2.6 DMA

The SPI module supports the use of DMA to speed up data communication, either by using DMA to fill the transmit buffer or by using DMA to pick up data from the receive buffer in a timely manner. DMA will pick up or send data in a timely manner using RXNE and TXE as signals. DMA can also operate in simplex or CRC mode.

### 16.2.7 Errors

- Master mode fault (MODF)

When the SPI is operating in NSS pin hardware management mode, an external pull-down of the NSS pin occurs; or in NSS pin software management mode, the SSI bit is cleared; or the SPE bit is cleared, causing the SPI to be shut down; or the MSTR bit is cleared and the SPI enters slave mode. If the ERRIE bit is already set, an interrupt is also generated. Steps to clear the MODF bit: First perform a read or write operation to R16_SPI1_STATR, and then write R16_SPI1_CTLR1.

- Overrun condition

If the host sends data and there is unread data in the receive buffer of the slave device, an overflow error occurs, the OVR bit is set, and an interrupt is also generated if ERRIE is set. Sending an overflow error should restart the current transmission. Reading the data register and then reading the status register will eliminate this bit.

- CRC error

When the received CRC word and the value of RXCRCR do not match, a CRC error will be generated and the CRCERR bit will be set.

### 16.2.8 Interrupts

The SPI module supports five interrupt sources, among which the TXE and RXNE events are set when the TXEIE and RXNEIE bits are set respectively. In addition to the above three errors will also generate interrupts, namely MODF, OVR and CRCERR, after enabling the ERRIE bit, these three errors will also generate error interrupts.

## 16.3 Register Description

Table 16-2 SPI-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_SPI1_CTLR1 | 0x40013000 | SPI control register 1 | 0x0000 |
| R16_SPI1_CTLR2 | 0x40013004 | SPI control register 2 | 0x0000 |
| R16_SPI1_STATR | 0x40013008 | SPI status register | 0x0002 |
| R16_SPI1_DATAR | 0x4001300C | SPI data register | 0x0000 |
| R16_SPI1_CRCR | 0x40013010 | SPI polynomial register | 0x0007 |

| | | | |
|---|---|---|---|
| R16_SPI1_RCRCR | 0x40013014 | SPI receive CRC register | 0x0000 |
| R16_SPI1_TCRCR | 0x40013018 | SPI transmit CRC register | 0x0000 |
| R16_SPI1_HSCR | 0x40013024 | SPI high-speed control register | 0x0000 |

### 16.3.1 SPI Control Register 1 (SPI1_CTLR1)

Offset address: 0x00

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BIDIMODE | BIDIOE | CRCEN | CRCNEXT | DFF | RXONLY | SSM | SSI | LSBFIRST | SPE | BR[2:0] | | | MSTR | CPOL | CPHA |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | BIDIMODE | RW | Unidirectional data mode enable bit:<br>1: Select 1-wire bidirectional mode;<br>0: Select 2-wire bidirectional mode. | 0 |
| 14 | BIDIOE | RW | Single-wire output enable bit, used in conjunction with BIDIMODE.<br>1: Enable output, transmit only;<br>0: Disable output, receive only. | 0 |
| 13 | CRCEN | RW | Hardware CRC checksum enable bit, this bit can only be written when SPE is 0. This bit can only be used in full-duplex mode.<br>1: Enable CRC calculation;<br>0: Disable CRC calculation. | 0 |
| 12 | CRCNEXT | RW | Transmit the value of the CRC register after the next data transfer. This should be set immediately after the last data is written to the data register.<br>1: Transmit the result of the CRC checksum;<br>0: Continue transmitting data from the data register. | 0 |
| 11 | DFF | RW | Data frame length bit, this bit can only be written when SPE is 0.<br>1: Use the 16-bit data length for sending and receiving;<br>0: Use 8-bit data length for sending and receiving. | 0 |
| 10 | RXONLY | RW | Receive only bit in 2-wire mode, this bit is used in conjunction with BIDIMODE. Set this bit to allow the device to receive only and not transmit.<br>1: Receive only, simplex mode;<br>0: Full-duplex mode. | 0 |
| 9 | SSM | RW | Chip select pin management bit, this bit determines whether the level of the NSS pin is controlled by hardware or software.<br>1: Software control of the NSS pin;<br>0: Hardware control of the NSS pin. | 0 |
| 8 | SSI | RW | Chip select pin control bit, with SSM set, this bit determines the level of the NSS pin.<br>1: NSS is high;<br>0: NSS is low. | 0 |
| 7 | LSBFIRST | RW | Frame format control bit. It is not possible to modify this bit during communication.<br>1: LSB is transmitted first;<br>0: MSB is transmitted first.<br>*Note: LSB is only supported by SPI as host.* | 0 |
| 6 | SPE | RW | SPI enable bit.<br>1: SPI enabled; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: SPI disabled. | |
| [5:3] | BR[2:0] | RW | Modified during communication.<br>000: FHCLK/2; 001: FHCLK/4;<br>010: FHCLK/8; 011: FHCLK/16;<br>100: FHCLK/32; 101: FHCLK/64;<br>110: FHCLK/128; 111: FHCLK/256.<br>*Note: This bit applies only if the HSRXEN bit is 0; when the HSRXEN bit is 1, the SCK frequency is FHCLK/(BR+2).* | 0 |
| 2 | MSTR | RW | Master-slave setting bit, this bit cannot be modified during communication.<br>1: Configured as a master device;<br>0: Configured as a slave device. | 0b |
| 1 | CPOL | RW | Clock polarity selection bit, this bit must not be modified during communication.<br>1: SCK is held high in the idle state;<br>0: SCK is held low in the idle state. | 0 |
| 0 | CPHA | RW | Clock phase set bit, this bit must not be modified during communication.<br>1: Data sampling starts from the second clock edge;<br>0: Data sampling starts from the first clock edge. | 0 |

### 16.3.2 SPI Control Register 2 (SPI1_CTLR2)

Offset address: 0x04

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ODEN | Reserved | | | | | | | TXEIE | RXNEIE | ERRIE | Reserved | | SSOE | TXDMAEN | RXDMAEN |

Control Register 2

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | ODEN | RW | Open-drain output enable for SPI pins:<br>1: Enable NSS, SCK, MOSI open-drain output, which can be used to support low-voltage SPI signals;<br>0: Disable SPI open-drain output, default push-pull. | 0 |
| [14:8] | Reserved | RO | Reserved | 0 |
| 7 | TXEIE | RW | Transmit buffer air break enable bit. Set this bit to allow an interrupt to be generated when TXE is set. | 0 |
| 6 | RXNEIE | RW | Receive buffer non-air break enable bit. Set this bit to allow an interrupt to be generated when RXNE is set. | 0 |
| 5 | ERRIE | RW | Error interrupt enable bit. Set this bit to allow an interrupt to be generated if an error (CRCERR, OVR, MODF) is generated. | 0 |
| [4:3] | Reserved | RO | Reserved | 0 |
| 2 | SSOE | RW | SS output enable. Disabling SS output can work in multi-master mode.<br>1: Enable the SS output;<br>0: Disable SS output in master mode. | 0 |
| 1 | TXDMAEN | RW | Transmit buffer DMA enable bit.<br>1: Enable transmit buffer DMA;<br>0: Disable transmit buffer DMA. | 0 |

| | | | | |
|---|---|---|---|---|
| 0 | RXDMAEN | RW | Receive buffer DMA enable bit.<br>1: Enable receive buffer DMA;<br>0: Disable receive buffer DMA. | 0 |

### 16.3.3 SPI Status Register (SPI1_STATR)

Offset address: 0x08

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | BSY | OVR | MODF | CRCERR | Reserved | | TXE | RXNE |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved | 0 |
| 7 | BSY | RO | Busy flag bit, this bit is set or reset by hardware.<br>1: SPI is communicating, or the transmit buffer is not empty;<br>0: SPI is not communicating. | 0 |
| 6 | OVR | RW0 | Overflow flag bit, this bit is set by hardware and reset by software.<br>1: An overflow error has occurred;<br>0: No overflow error has occurred. | 0 |
| 5 | MODF | RO | Mode error flag bit, this bit is set by hardware and reset by software.<br>1: A mode error has occurred;<br>0: No mode error has occurred. | 0 |
| 4 | CRCERR | RW0 | CRC error flag bit, this bit is set by hardware and reset by software.<br>1: The received CRC value does not match the value of the RCRCR;<br>0: The received CRC value is the same as the value of the RRCR. | 0 |
| [3:2] | Reserved | RO | Reserved | 0 |
| 1 | TXE | RO | Transmit buffer empty flag bit:<br>1: The transmit buffer is empty;<br>0: The transmit buffer is not empty. | 1 |
| 0 | RXNE | RO | Receive buffer not empty flag bit:<br>1: Receive buffer is not empty;<br>0: Receive buffer is empty. | 0 |

### 16.3.4 SPI Data Register (SPI1_DATAR)

Offset address: 0x0C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DR[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | DR[15:0] | RW | Data register. The data registers are used to store received data or pre-store data to be sent out, so the reading and writing of data registers actually corresponds to the operation of different areas, | 0 |

| | | | | |
|---|---|---|---|---|
| | | | where reads correspond to the receive buffer and writes to the send buffer. Data can be received and sent in 8 or 16 bits, and it is necessary to determine how many bits of data to use before transmission. When using 8 bits for data transmission, only the lower 8 bits of the data register are used and the higher 8 bits are forced to 0 for reception; using a 16-bit data structure causes all 16 bits of the data register to be used. | |

### 16.3.5 SPI Polynomial Register (SPI1_CRCR)

Offset address: 0x10

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CRCPOLY[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | CRCPOLY[15:0] | RW | CRC polynomial. This field defines the polynomial used in the CRC calculation. | 7 |

### 16.3.6 SPI Receive CRC Register (SPI1_RCRCR)

Offset address: 0x14

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RXCRC[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | RXCRC[15:0] | RO | Receive CRC value. Stores the result of the calculated CRC checksum of the received byte. Setting CRCEN resets this register. The calculation uses the polynomial used by CRCPOLY; only the lower 8 bits are involved in the calculation in 8-bit mode and all 16 bits are involved in the calculation in 16-bit mode. This register needs to be read when BSY is 0. | 0 |

### 16.3.7 SPI Transmit CRC Register (SPI1_TCRCR)

Offset address: 0x18

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TXCRC[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | TXCRC[15:0] | RO | Transmit CRC value. Stores the result of the calculated CRC checksum of the bytes that have been sent out. Setting CRCEN resets this register. The calculation is done using the polynomial used in CRCPOLY. 8-bit mode only the lower 8 bits are involved in the calculation, 16-bit mode all 16 bits are involved. This register needs to be read when BSY is 0. | 0 |

### 16.3.8 SPI High-speed Control Register (SPI1_HSCR)

Offset address: 0x24

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | HSRXEN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:1] | Reserved | RO | Reserved | 0 |
| 0 | HSRXEN | WO | Read enable in SPI high speed mode (CLK greater than or equal to 36MHz). This mode is only valid when the clock is divided by 2 (i.e. BR = 000 in the CTLR1 register). This bit is not readable.<br>1: Enable high-speed read mode;<br>0: Disable high-speed read mode. | 0 |

# Chapter 17 Operational Amplifiers (OPA) and Comparators (CMP)

The module contains 2 independently configurable operational amplifiers (OPA or PGA) and 3 independently configurable voltage comparators (CMP), where the operational amplifiers (OPA or PGA) support gain selection and can also be adapted for use as voltage comparators.

The inputs and outputs of each op-amp are connected to I/O ports with selectable input pins or gain, and the output pins can be optionally configured to the general purpose I/O port or alternate as I/O for the ADC sample channels, supporting the amplification of small external analogue signals into the ADC for small signal ADC conversion.

The input and output of each voltage comparator are connected to the I/O port and the input pins are selectable and the output pins can be optionally configured to the general purpose I/O port or alternate as TIM internal sampling channels (Without taking up I/O pins).

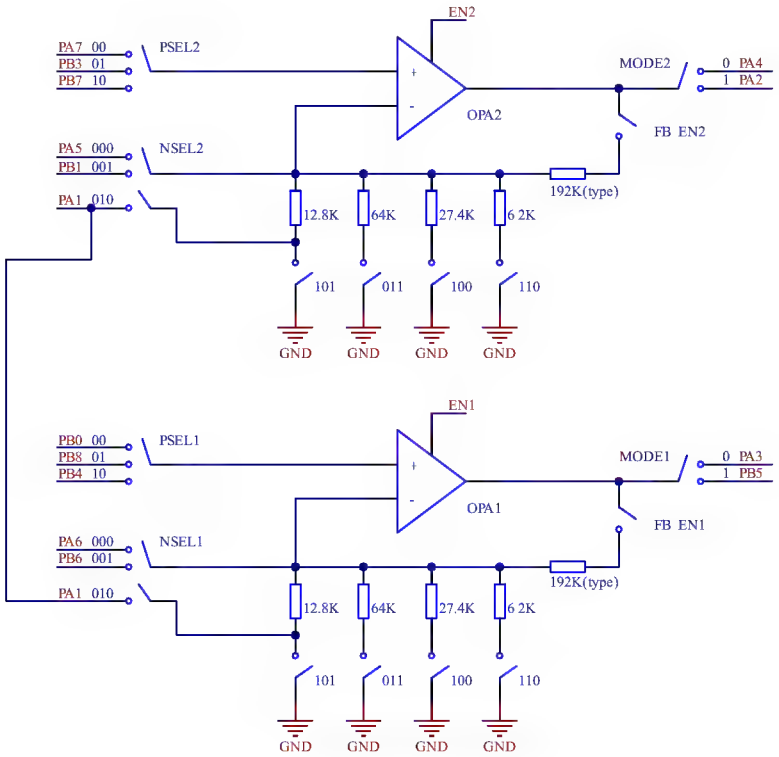
## 17.1 Main Features

- OPA input pin or channel selectable
- OPA output pins selectable for general purpose I/O ports or ADC sampling channels
- OPA supports positive input polling
- OPA supports PGA gain selection
- CMP input pin selectable, negative input channel selectable common pins
- CMP output pins selectable for general purpose I/O port or TIM internal sample channel
- 1 interrupt vector

## 17.2 Function Description

### 17.2.1 OPA

Configuring MODEx in the OPA_CTLR1 register selects the output channel of the OPAx as an ADC sample channel or a normal I/O port, configuring PSELx in the OPA_CTLR1 register selects the positive input pin of the OPAx, and configuring NSELx in the OPA_CTLR1 register selects the negative input channel of the OPAx or the gain when used as a PGA.




### 17.2.2 Polling of Positive OPA Inputs

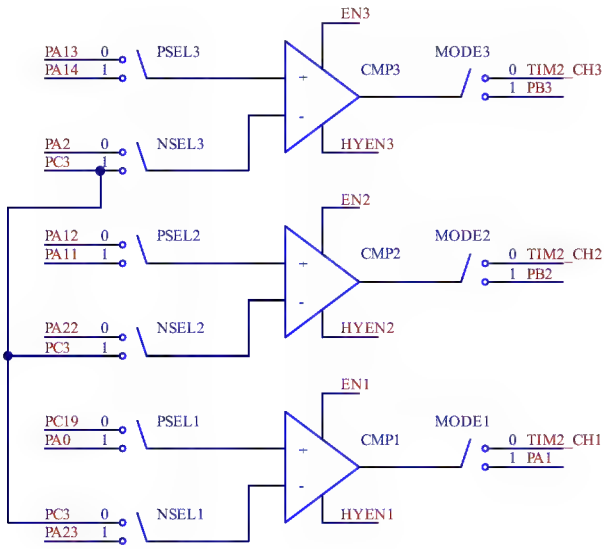
The polling function of OPA can be implemented by selecting OPA_P0/OPA_P1/OPA_P2 at regular intervals and selecting all P-terminals in turn; the polling OPAx can be selected by configuring the POLL_EN bit in the OPA_CFGR register.

The number of channels to be polled can be configured via POLLx_NUM in the OPA_CFGR register, and the polling interval can be configured via POLL_VLU in the OPA_CFGR register.

### 17.2.3 CMP

To enable the corresponding CMPx, set ENx in the OPA_CTLR2 register. Configure MODEx in the OPA_CTLR2 register to select the output channel of the CMPx as a normal I/O port or an internal timer channel. Configure PSELx in the OPA_CTLR2 register to select the positive input pin of the CMPx, and configure NSELx in the OPA_CTLR2 register to select the negative input pin of the CMPx.

*Note: For detailed input and output pins of each OPA and CMP, refer to the pin descriptions in the datasheet.*




## 17.3 Register Description

Table 17-1 OPA-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_OPA_CFGR1 | 0x40026000 | OPA configuration register 1 | 0x0080 |
| R16_OPA_CFGR2 | 0x40026002 | OPA configuration register 2 | 0x0000 |
| R32_OPA_CTLR1 | 0x40026004 | OPA control register 1 | 0x81D801D8 |
| R32_OPA_CTLR2 | 0x40026008 | OPA control register 2 | 0x80000000 |
| R32_OPA_KEY | 0x4002600C | OPA unlock key register | 0xXXXXXXXX |
| R32_CMP_KEY | 0x40026010 | CMP unlock key register | 0xXXXXXXXX |
| R32_POLL_KEY | 0x40026014 | POLL unlock key register | 0xXXXXXXXX |

### 17.3.1 OPA Configuration Register 1 (OPA_CFGR1)

Offset address: 0x00

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | IF_CNT | IF_OUT2 | IF_OUT1 | NMI_EN | IE_CNT | IE_OUT2 | IE_OUT1 | POLL_LOCK | BKIN_SEL | RST_EN2 | RST_EN1 | BKIN_EN2 | BKIN_EN1 | POLL_EN2 | POLL_EN1 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | Reserved | RO | Reserved | 0 |
| 14 | IF_CNT | RW0 | Interrupt flag for the end of the OPA polling interval:<br>1: End of polling interval;<br>0: Invalid.<br>Write 0 clears, write 1 invalid. | 0 |
| 13 | IF_OUT2 | RW0 | Interrupt flag for polling to an OPA2 output high:<br>1: Poll to OPA2 output high;<br>0: Invalid.<br>Write 0 cleared, write 1 invalid. | 0 |
| 12 | IF_OUT1 | RW0 | Interrupt flag for polling to an OPA1 output high:<br>1: Poll to OPA1 output high;<br>0: Invalid.<br>Write 0 cleared, write 1 invalid. | 0 |

| | | | | |
|---|---|---|---|---|
| 11 | NMI_EN | RW | OPA connection NMI interrupt enable:<br>1: On;<br>0: Off. | 0 |
| 10 | IE_CNT | RW | OPA end-of-polling-interval interrupt enable:<br>1: Interrupt enable on;<br>0: Interrupt enable off. | 0 |
| 9 | IE_OUT2 | RW | OPA2 interrupt enable:<br>1: Interrupt enable on;<br>0: Interrupt enable off. | 0 |
| 8 | IE_OUT1 | RW | OPA1 interrupt enable:<br>1: Interrupt enable on;<br>0: Interrupt enable off. | 0 |
| 7 | POLL_LOCK | W1 | POLL lock (Write 1 is locked, write 0 is invalid):<br>1: Locked, no other bits of the configuration register can be written to;<br>0: Unlocked, write operations can be performed on other bits of the configuration register. | 1 |
| 6 | BKIN_SEL | RW | Timer selection for brake input connection (Two timers swapped):<br>Outputs to timer 1:<br>1: OPA2 output;<br>0: OPA1 output.<br>Output to timer 2:<br>1: OPA1 output;<br>0: OPA2 output.<br>*Note: The brake signal can only be active high and requires the TIMx_BDTR register BKP position 1.* | 0 |
| 5 | RST_EN2 | RW | OPA2 reset system enable:<br>1: Reset enable on;<br>0: Reset enable off. | 0 |
| 4 | RST_EN1 | RW | OPA1 reset system enable:<br>1: Reset enable on;<br>0: Reset enable off. | 0 |
| 3 | BKIN_EN2 | RW | Timer's brake input source OPA2 enable:<br>1: OPA2 brake enable on;<br>0: OPA2 brake enable off. | 0 |
| 2 | BKIN_EN1 | RW | Timer's brake input source OPA1 enable:<br>1: OPA1 brake enable on;<br>0: OPA1 brake enable off. | 0 |
| 1 | POLL_EN2 | RW | OPA2 positive polling enable:<br>1: Positive polling enable on;<br>0: Positive polling enable off. | 0 |
| 0 | POLL_EN1 | RW | OPA1 positive polling enable:<br>1: Positive polling enable on;<br>0: Positive polling enable off. | 0 |

### 17.3.2 OPA Configuration Register 2 (OPA_CFGR2)

Offset address: 0x02

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | POLL2_NUM | | POLL1_NUM | | POLL_VLU | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:13] | Reserved | RO | Reserved | 0 |
| [12:11] | POLL2_NUM | WO | Configure the number of positive ends polled by OPA2:<br>00: 1, O2P0;<br>01: 2, O2P0+O2P1;<br>10: 3, O2P0+O2P1+O2P2;<br>11: Reserved. | 0 |
| [10:9] | POLL1_NUM | WO | Configure the number of positive ends polled by OPA1:<br>00: 1, O1P0;<br>01: 2, O1P0+O1P1;<br>10: 3, O1P0+O1P1+O1P2;<br>11: Reserved. | 0 |
| [8:0] | POLL_VLU | RW | Configure the OPA positive end polling interval:<br>Polling interval = (POLL_VLU+1)*1us. | 0 |

### 17.3.3 OPA Control Register 1 (OPA_CTLR1)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OPA_LOCK | Reserved | | | | | | NSEL2 | | | FB_EN2 | PSEL2 | | Reserved | MODE2 | EN2 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | NSEL1 | | | FB_EN1 | PSEL1 | | Reserved | MODE1 | EN1 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 31 | OPA_LOCK | W1 | OPA lock (Write 1 to lock, write 0 to invalidate):<br>1: OPA locked;<br>0: OPA unlocked. | 1 |
| [30:25] | Reserved | RO | Reserved. | 0 |
| [24:22] | NSEL2 | RW | OPA2 negative input channel, and gain selection when used as PGA:<br>000: PA5;<br>001: PB1;<br>010: PA1, PGA mode with external pins, internal 16x amplification, externally settable negative reference voltage, string resistor to turn down gain;<br>011: PGA mode without external pins, fixed approx. 4x amplification;<br>100: PGA mode without external pins, fixed approx. 8x amplification;<br>101: PGA mode without external pin, fixed approx. 16x amplification;<br>110: PGA mode without external pins, fixed approx. 32x amplification;<br>111: switched off. | 111 |
| 21 | FB_EN2 | RW | OPA2 internal feedback resistor enable:<br>1: Enable, used as PGA2, feedback resistor approx. 192 kΩ;<br>0: Disable.<br>*Note: This bit must be set to 1 when the NSEL2 control bit above is in PGA mode.* | 0 |
| [20:19] | PSEL2[1:0] | RW | OPA2 positive input selection: | 11 |

| | | | | |
|---|---|---|---|---|
| | | | 00: PA7;<br>01: PB3;<br>10: PB7;<br>11: Forbidden. | |
| 18 | Reserved | RO | Reserved. | 0 |
| 17 | MODE2 | RW | OPA2 output channel selection:<br>1: Output channel is PA2;<br>0: Output channel is PA4. | 0 |
| 16 | EN2 | RW | OPA2 enable:<br>1: OPA2 enabled;<br>0: OPA2 disabled. | 0 |
| [15:9] | Reserved | RO | Reserved. | 0 |
| [8:6] | NSEL1 | RW | OPA1 negative input channel, and gain selection when used as a PGA<br>000: PA6;<br>001: PB6;<br>010: PA1, PGA mode with external pins, internal 16x amplification, externally settable negative reference voltage, string resistor to turn down gain;<br>011: PGA mode without external pin, fixed approx. 4x amplification;<br>100: PGA mode without external pin, fixed approx. 8x amplification;<br>101: PGA mode without external pin, fixed approx. 16x amplification;<br>110: PGA mode without external pins, fixed approx. 32x amplification;<br>111: Off. | 111 |
| 5 | FB_EN1 | RW | OPA1 internal feedback resistor enable:<br>1: Enable, used as PGA1, feedback resistor approx. 192 kΩ;<br>0: Disable.<br>*Note: This bit must be set to 1 when the NSEL1 control bit above is in PGA mode.* | 0 |
| [4:3] | PSEL1[1:0] | RW | OPA1 positive input selection:<br>00: PB0;<br>01: PB8;<br>10: PB4;<br>11: Prohibited. | 11 |
| 2 | Reserved | RO | Reserved. | 0 |
| 1 | MODE1 | RW | OPA1 output channel selection:<br>1: Output channel is PB5;<br>0: Output channel is PA3. | 0 |
| 0 | EN1 | RW | OPA1 enable:<br>1: OPA1 enabled;<br>0: OPA1 disabled. | 0 |

### 17.3.4 OPA Control Register 2 (OPA_CTLR2)

Offset address: 0x08

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CMP_LOCK | Reserved | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

| Reserved | HYEN3 | PSEL3 | NSEL3 | MODE3 | EN3 | HYEN2 | PSEL2 | NSEL2 | MODE2 | EN2 | HYEN1 | PSEL1 | NSEL1 | MODE1 | EN1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 31 | CMP_LOCK | W1 | CMP lock (Write 1 is locked, write 0 is invalid):<br>1: CMP locked;<br>0: unlock CMP. | 1 |
| [30:15] | Reserved | RO | Reserved. | 0 |
| 14 | HYEN3 | RW | CMP3 comparator hysteresis function selection<br>1: On;<br>0: Off. | 0 |
| 13 | PSEL3 | RW | CMP3 positive input channel selection:<br>1: PA14;<br>0: PA13. | 0 |
| 12 | NSEL3 | RW | CMP3 negative input channel selection:<br>1: PC3;<br>0: PA2. | 0 |
| 11 | MODE3 | RW | CMPA3 output channel selection:<br>1: Output channel is PB3;<br>0: Output channel is internal channel TIM2_CH3. | 0 |
| 10 | EN3 | RW | CMP3 enable:<br>1: Enable CMP3;<br>0: Disable CMP3. | 0 |
| 9 | HYEN2 | RW | CMP2 comparator hysteresis function selection<br>1: On;<br>0: Off. | 0 |
| 8 | PSEL2 | RW | CMP2 positive input channel selection:<br>1: PA11;<br>0: PA12. | 0 |
| 7 | NSEL2 | RW | CMP2 negative input channel selection:<br>1: PC3;<br>0: PA22. | 0 |
| 6 | MODE2 | RW | CMPA2 output channel selection:<br>1: Output channel is PB2;<br>0: Output channel is internal channel TIM2_CH2. | 0 |
| 5 | EN2 | RW | CMP2 enable：<br>1: Enable CMP2;<br>0: Disable CMP2. | 0 |
| 4 | HYEN1 | RW | CMP1 comparator hysteresis function selection<br>1: On;<br>0: Off. | 0 |
| 3 | PSEL1 | RW | CMP1 positive input channel selection:<br>1: PA0;<br>0: PC19. | 0 |
| 2 | NSEL1 | RW | CMP1 negative input channel selection:<br>1: PA23;<br>0: PC3. | 0 |
| 1 | MODE1 | RW | CMPA1 output channel selection:<br>1: Output channel is PA1;<br>0: Output channel is internal channel TIM2_CH1. | 0 |
| 0 | EN1 | RW | CMP1 enable：<br>1: Enable CMP1;<br>0: Disable CMP1. | 0 |

### 17.3.5 OPA Unlock Key Register (OPA_KEY)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OPA_KEY[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| OPA_KEY[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | OPA_KEY[31:0] | RW | OPA keys, used to enter OPA unlock keys include:<br>KEY1 = 0x45670123;<br>KEY2 = 0xCDEF89AB. | X |

### 17.3.6 CMP Unlock Key Register (CMP_KEY)

Offset address: 0x10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CMP_KEY[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CMP_KEY[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | CMP_KEY[31:0] | RW | CMP keys, used to enter the CMP unlock key include:<br>KEY1 = 0x45670123;<br>KEY2 = 0xCDEF89AB. | X |

### 17.3.7 POLL Unlock Key Register (POLL_KEY)

Offset address: 0x14

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| POLL_KEY[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| POLL_KEY[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | POLL_KEY[31:0] | RW | POLL keys, used to enter the POLL unlock key include:<br>KEY1 = 0x45670123;<br>KEY2 = 0xCDEF89AB。 | X |

# Chapter 18 USB Full-speed Host/Device Controller (USBFS)

## 18.1 Introduction to USB Controller

The chip is embedded with a USB controller and transceiver with the following features:

- Dual role controller supporting USB Host function and USB Device function.
- Comply with the On-The-GoSupplement to the USB2.0 specification and supports USB2.0 full-speed 12Mbps or low-speed 1.5Mbps in both host and device modes.
- Support software HNP and SRP protocols.
- Support USB control transfers, bulk transfers, interrupt transfers, and synchronous/real-time transfers.
- Support packets up to 64 bytes, built-in FIFO, interrupt and DMA support.

## 18.2 Register Description

USB related registers are divided into 3 sections, some of which are alternate in host and device mode.

- USB Global Register
- USB Device Control Register
- USB Host Control Register

### 18.2.1 Global Register Description

Table 18-1 USBFS-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_BASE_CTRL | 0x40023400 | USB control register | 0x06 |
| R8_INT_EN | 0x40023402 | USB interrupt enable register | 0x00 |
| R8_DEV_ADDR | 0x40023403 | USB device address register | 0x00 |
| R8_MIS_ST | 0x40023405 | USB miscellaneous status register | 0xXX |
| R8_INT_FG | 0x40023406 | USB interrupt flag register | 0x20 |
| R8_INT_ST | 0x40023407 | USB interrupt status register | 0xXX |
| R16_RX_LEN | 0x40023408 | USB receive length register | 0xXX |
| R8_UEP4_1_MOD | 0x4002340C | Endpoint 1 and 4 mode control registers | 0x00 |
| R8_UEP2_3_MOD | 0x4002340D | Endpoint 2 and 3 mode control registers | 0x00 |
| R8_UEP567_MOD | 0x4002340E | Endpoint 5, 6 and 7 mode control registers | 0x00 |

#### 18.2.1.1 USB Control Register (R8_BASE_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UC_HOST_MODE | RW | USB operating mode selection bits:<br>1: Host mode (HOST);<br>0: Device mode (DEVICE). | 0 |
| 6 | RB_UC_LOW_SPEED | RW | USB low speed enable bit:<br>1: Low speed enabled;<br>0: Low speed disabled. | 0 |
| [5:4] | RB_SYS_MODE | RW | Test mode in host mode. | 0 |
| 3 | RB_UC_INT_BUSY | RW | USB transfer completion interrupt flag not cleared to zero Auto pause enable bit:<br>1: Auto pause before interrupt flag UIF_TRANSFER is cleared, auto answer busy NAK in device mode, auto pause subsequent transfers in host mode; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: No pause. | |
| 2 | RB_UC_RST_SIE | RW | USB protocol processor software reset control bits:<br>1: Forced reset of the USB protocol processor (SIE), requiring a software reset;<br>0: No reset. | 1 |
| 1 | RB_UC_CLR_ALL | RW | USB FIFO and interrupt flag clear: 1<br>1: Clears the USB interrupt flag and FIFO, requires software clearing;<br>0: Not cleared. | 1 |
| 0 | RB_UC_DMA_EN | RW | Enables DMA for USB, this bit must be set to 1 in normal transfer mode:<br>1: Enables the DMA function and the DMA interrupt;<br>0: Disables DMA. | 0 |

#### 18.2.1.2 USB Interrupt Enable Register (R8_INT_EN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved. | 0 |
| 6 | RB_UIE_DEV_NAK | RW | USB device mode, receive NAK interrupt:<br>1: Enable the corresponding interrupt;<br>0: Disable the corresponding interrupt. | 0 |
| 5 | RB_UID_1_WIRE | RW | USB 1-wire mode enable:<br>1: Enable; 0: Disable. | 0 |
| 4 | RB_UIE_FIFO_OV | RW | FIFO overflow interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 3 | RB_UIE_SOF_ACT | RW | USB host mode, SOF receive completion interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 2 | RB_UIE_SUSPEND | RW | USB bus suspend or wakeup event interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 1 | RB_UIE_TRANSFER | RW | USB transfer (Excluding SETUP transaction) completion interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 0 | RB_UIE_DETECT | RW | USB host mode, USB device connect or disconnect event interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| | RB_UIE_BUS_RST | RW | USB device mode, USB bus reset event interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |

#### 18.2.1.3 USB Device Address Register (R8_ DEV_ADDR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved. | 0 |
| [6:0] | MASK_USB_ADDR | RW | Host mode is the address or HUB address of the USB device currently in operation;<br>Device mode: the address of that USB itself. | 0 |

#### 18.2.1.4 USB Miscellaneous Status Register (R8_MIS_ST)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UMS_SOF_PRES | RO | SOF packet presage status in USB host mode: | X |

| | | | | |
|---|---|---|---|---|
| | | | 1: SOF packet is about to be transmitted, if there are other USB packets at this time, it will be automatically delayed;<br>0: No SOF packet is transmitted. | |
| 6 | RB_UMS_SOF_ACT | RO | SOF packet transfer status in USB host mode:<br>1: SOF packet is being transmitted;<br>0: Transmit complete or idle. | X |
| 5 | RB_UMS_SIE_FREE | RO | USB protocol handler free:<br>1: The protocol device is free;<br>0: Busy, USB transfer in progress. | 1 |
| 4 | RB_UMS_R_FIFO_RDY | RO | USB receive FIFO data ready:<br>1: The receive FIFO is not empty;<br>0: The receive FIFO is empty. | 0 |
| 3 | RB_UMS_BUS_RST | RO | USB bus reset:<br>1: The current USB bus is in reset state;<br>0: The current USB bus is in a non-reset state. | X |
| 2 | RB_UMS_SUSPEND | RO | USB suspend:<br>1: The USB bus is in a suspended state, and there is no USB activity for a period of time;<br>0: The USB bus is in a non-suspend state. | 0 |
| 1 | RB_UMS_DM_LEVEL | RO | In USB host mode, level state of the DM pin when the device is just connected to the USB port is used to judge the speed:<br>1: High level/low-speed;<br>0: Low level/full-speed. | 0 |
| 0 | RB_UMS_DEV_ATTACH | RO | USB device attach status for the port in USB host mode:<br>1: The port has been connected to a USB device;<br>0: The port has no USB device connected. | 0 |

#### 18.2.1.5 USB Interrupt Flag Register (R8_INT_FG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_U_IS_NAK | RO | NAK response interrupt flag bit in USB device mode, same as RB_U_IS_NAK:<br>1: Response to NAK during the current USB transfer;<br>0: No event. | 0 |
| 6 | TOG_MATCH_SYNC | RO | Toggle of the received packet with the set expectation value after the USB transaction reception is completed.<br>Matching status bits:<br>1: Toggle matched;<br>0: Toggle does not match. | 0 |
| 5 | RB_U_SIE_FREE | RO | USB Protocol Handler free:<br>1: USB idle;<br>0: Busy, USB transfer in progress. | 1 |
| 4 | RB_UIF_FIFO_OV | RW | USB FIFO overflow interrupt flag, write 1 to clear:<br>1: FIFO overflow trigger;<br>0: No event. | 0 |
| 3 | RB_UIF_HST_SOF | RW | SOF timer interrupt flag in USB host mode, write 1 to clear: | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: SOF packet transmission completion trigger;<br>0: No event. | |
| 2 | RB_UIF_SUSPEND | RW | USB bus suspend or wake-up event interrupt flag, write 1 to clear:<br>1: Triggered by USB suspend event or wake-up event;<br>0: No event. | 0 |
| 1 | RB_UIF_TRANSFER | RW | USB transfer completion interrupt flag, write 1 to clear:<br>1: Trigger when a USB transfer is completed;<br>0: No event. | 0 |
| 0 | RB_UIF_DETECT | RW | In USB host mode, USB device connect or disconnect event interrupt flag, write 1 to clear:<br>1: Detected USB device connection or disconnection trigger;<br>0: No event. | 0 |
| | RB_UIF_BUS_RST | RW | In USB device mode, USB bus reset event interrupt flag bit, write 1 to clear:<br>1: USB bus reset event trigger;<br>0: No event. | 0 |

**18.2.1.6 USB Interrupt Status Register (R8_INT_ST)**

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | SETUP_ACT | RO | SETUP transaction complete<br>1: SETUP transaction complete;<br>0: No event. | 0 |
| 6 | RB_UIS_TOG_OK | RO | Toggle of the received packet after the USB transaction reception is completed with the set expectation<br>Match status bits:<br>1: Toggle matched;<br>0: Toggle does not match. | 0 |
| [5:4] | MASK_UIS_TOKEN | RO | In device mode, the token PID identifier of the current USB transfer transaction. | XXb |
| [3:0] | MASK_UIS_ENDP | RO | In device mode, the endpoint number of the current USB transfer transaction. | XXXXb |
| | MASK_UIS_H_RES | RO | In host mode, the response PID identifier of the current USB transmission transaction, 0000 means the device has no response or timed out; other values indicate the response PID. | XXXXb |

*Note: MASK_UIS_TOKEN is used in USB device mode to identify the token PID of the current USB transmission transaction: 00 for OUT packets; 01 reserved; 10 for IN packets; 11 for SETUP packets.*
*MASK_UIS_H_RES is only valid in host mode. In host mode, if the host sends an OUT/SETUP token packet, this PID is either a handshake packet ACK/NAK/STALL, or the device has no answer/timeout. If the host sends an IN token packet, this PID is either the packet PID (DATA0/DATA1) or the handshake packet PID.*

**18.2.1.7 USB Receive Length Register (R16_RX_LEN)**

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:0] | USB_RX_LEN | RO | The current number of data bytes received by the USB endpoint | X |

#### 18.2.1.8 Endpoint 1 and 4 mode controller (R8_UEP4_1_MOD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | EP1_R_EN | RW | Endpoint 1 receive enable. | 0 |
| 6 | EP1_T_EN | RW | Endpoint 1 transmit enable. | 0 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | EP1_BUF_MOD | RW | Endpoint 1 buffer mode control bit. | 0 |
| 3 | EP4_R_EN | RW | Endpoint 4 receive enable. | 0 |
| 2 | EP4_T_EN | RW | Endpoint 4 transmit enable. | 0 |
| [1:0] | Reserved | RO | Reserved | 0 |

The combination of EP4_R_EN and EP4_T_EN configures the data buffer mode for USB endpoints 0 and 4, as shown in the following table:

Table 18-2 Endpoint 0 and 4 buffer modes

| EP4_R_EN | EP4_T_EN | Description: Low to high with UEP0_DMA as starting address |
|---|---|---|
| 0 | 0 | Endpoint 0 single 64-byte send/receive common buffer (IN and OUT). |
| 1 | 0 | Endpoint 0 single 64-byte send/receive common buffer; Endpoint 4 single 64-byte receive buffer (OUT) |
| 0 | 1 | Endpoint 0 single 64-byte send/receive common buffer; Endpoint 4 single 64-byte send buffer (IN) |
| 1 | 1 | Endpoint 0 single 64-byte send/receive common buffer; Endpoint 4 single 64-byte receive buffer (OUT);<br>Endpoint 4 single 64-byte receive buffer (IN). The total 192 bytes are arranged as follows:<br>UEP0_DMA+0 address: the 64-byte start address of the endpoint 0 send/receive common buffer;<br>UEP0_DMA+64 address: 64-byte start address of the endpoint 4 receive buffer;<br>UEP0_DMA+128 address: endpoint 4 transmit buffer 64 bytes start address. |

#### 18.2.1.9 Endpoint 2 and 3 mode controller (R8_UEP2_3_MOD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | EP3_R_EN | RW | Endpoint 3 receive enable. | 0 |
| 6 | EP3_T_EN | RW | Endpoint 3 transmit enable. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | EP3_BUF_MOD | RW | Endpoint 3 buffer mode control bit. | 0 |
| 3 | EP2_R_EN | RW | Endpoint 2 receive enable. | 0 |
| 2 | EP2_T_EN | RW | Endpoint 2 transmit enable. | 0 |
| 1 | Reserved | RO | Reserved. | 0 |
| 0 | EP2_BUF_MOD | RW | Endpoint 2 buffer mode control bit. | 0 |

#### 18.2.1.10 Endpoint 5, 6 and 7 mode controller (R8_UEP567_MOD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:6] | Reserved | RO | Reserved. | 0 |
| 5 | EP7_R_EN | RW | Endpoint 7 receive enable. | 0 |
| 4 | EP7_T_EN | RW | Endpoint 7 transmit enable. | 0 |
| 3 | EP6_R_EN | RW | Endpoint 6 receive enable. | 0 |
| 2 | EP6_T_EN | RW | Endpoint 6 transmit enable. | 0 |
| 1 | EP5_R_EN | RW | Endpoint 5 receive enable. | 0 |
| 0 | EP5_T_EN | RW | Endpoint 5 transmit enable. | 0 |

### 18.2.2 USB Device Register Description

The USBFS module provides a total of 8 sets of bidirectional endpoints from endpoints 0-7 in USB device mode, and the maximum packet length for all endpoints is 64 bytes.

- Endpoint 0 is the default endpoint, which supports control transmission and shares a 64-byte data buffer between transmit and receive.
- Endpoints 1-7 each include a transmit endpoint IN and a receive endpoint OUT, each with a separate data buffer for transmit and receive, and support bulk, interrupt and real-time/synchronous transfers.
- Endpoint 0 has a separate DMA address, shared between transmit and receive, and endpoints 1-7 each have a DMA address for transmit and receive. The data buffer mode can be set to double buffer or single buffer by entering the R32_UEPn_BUF_MOD register. If the double buffer mode is used, the endpoint can only use single direction transmission.
- Each set of endpoints has transceiver control registers R8_UEPn_TX_CTRL, R8_UEPn_RX_CTRL and transmit length registers R16_UEPn_T_LEN and R32_UEPn_*¬_DMA (n=0~7), which are used to configure the synchronous trigger bits for that endpoint, the response to OUT transactions and IN transactions, and the length of the transmitted data. response, and the length of data sent, etc.

The USB bus pull-up resistor necessary as a USB device can be set by software at any time to enable or disable it. When a USB bus reset, USB bus hang or wake-up event is detected, or when the USB has successfully processed a data send or data receive, the USB protocol processor sets the corresponding interrupt flag and, if interrupt enable is on, also generates the corresponding interrupt request. The application can query and analyse the interrupt flag register R8_USB_INT_FG either directly or in the USB interrupt service program and process it accordingly according to RB_UIF_BUS_RST and RB_UIF_SUSPEND; and, if RB_UIF_TRANSFER is valid, then it also needs to continue to analyze the USB interrupt status register R8_USB_INT_ST and process accordingly according to the current endpoint number MASK_UIS_ENDP and the current transaction token PID identification MASK_UIS_TOKEN. If the synchronization trigger bit RB_UEP_R_TOG is set in advance for each endpoint's OUT transaction, then it can be judged by RB_U_TOG_OK or RB_UIS_TOG_OK whether the synchronization trigger bit of the currently received packet matches the synchronization trigger bit of that endpoint; if the data is synchronized, the data is valid; if the data is not synchronized, the data should be discarded. After each USB transmit or receive interrupt is processed, the corresponding endpoint's sync trigger bit should be correctly modified for the next sent packet or the next received packet to detect if it is synchronous; in addition, setting RB_UEP_T_TOG_AUTO or RB_UEP_R_TOG_AUTO enables the corresponding sync trigger bit to be automatically modified after a successful transmit or receive (Flip or self-decrease).

The data ready to be sent by each endpoint is in its own buffer, and the length of the data ready to be sent is set independently in R16_UEPn_T_LEN; the data received by each endpoint is in its own buffer, but the length of the data received is in the USB receive length register R16_USB_RX_LEN, which can be distinguished according to the current endpoint number during a USB receive interrupt.

Table 18-3 Device-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_UDEV_CTRL | 0x40023401 | USB device physical port control register | 0xX0 |
| R32_UEP0_DMA | 0x40023410 | Start address of endpoint 0 buffer | 0x00000000 |
| R32_UEP1_DMA | 0x40023414 | Start address of endpoint 1 buffer | 0x00000000 |
| R32_UEP2_DMA | 0x40023418 | Start address of endpoint 2 buffer | 0x00000000 |
| R32_UEP3_DMA | 0x4002341C | Start address of endpoint 3 buffer | 0x00000000 |
| R32_UEP0_CTRL | 0x40023420 | Endpoint 0 transmit length and control registers | 0x00000000 |

| | | | |
|---|---|---|---|
| R16_UEP0_TX_LEN | 0x40023420 | Endpoint 0 transmit length register | 0x0000 |
| R16_UEP0_CTRL_H | 0x40023422 | Endpoint 0 control register | 0x0000 |
| R32_UEP1_CTRL | 0x40023424 | Endpoint 1 transmit length and control registers | 0x00000000 |
| R16_UEP1_TX_LEN | 0x40023424 | Endpoint 1 transmit length register | 0x0000 |
| R16_UEP1_CTRL_H | 0x40023426 | Endpoint 1 control register | 0x0000 |
| R32_UEP2_CTRL | 0x40023428 | Endpoint 2 transmit length and control registers | 0x00000000 |
| R16_UEP2_TX_LEN | 0x40023428 | Endpoint 2 transmit length register | 0x0000 |
| R16_UEP2_CTRL_H | 0x4002342A | Endpoint 2 control register | 0x0000 |
| R32_UEP3_CTRL | 0x4002342C | Endpoint 3 transmit length and control registers | 0x00000000 |
| R16_UEP3_TX_LEN | 0x4002342C | Endpoint 3 transmit length register | 0x0000 |
| R16_UEP3_CTRL_H | 0x4002342E | Endpoint 3 control register | 0x0000 |
| R32_UEP4_CTRL | 0x40023430 | Endpoint 4 transmit length and control registers | 0x00000000 |
| R16_UEP4_TX_LEN | 0x40023430 | Endpoint 4 transmit length register | 0x0000 |
| R16_UEP4_CTRL_H | 0x40023432 | Endpoint 4 control register | 0x0000 |
| R32_UEP5_DMA | 0x40023454 | Start address of endpoint 5 buffer | 0x00000000 |
| R32_UEP6_DMA | 0x40023458 | Start address of endpoint 6 buffer | 0x00000000 |
| R32_UEP7_DMA | 0x4002345C | Start address of endpoint 7 buffer | 0x00000000 |
| R32_UEP5_CTRL | 0x40023464 | Endpoint 5 transmit length and control registers | 0x00000000 |
| R16_UEP5_TX_LEN | 0x40023464 | Endpoint 5 transmit length register | 0x0000 |
| R16_UEP5_CTRL_H | 0x40023466 | Endpoint 5 control register | 0x0000 |
| R32_UEP6_CTRL | 0x40023468 | Endpoint 6 transmit length and control registers | 0x00000000 |
| R16_UEP6_TX_LEN | 0x40023468 | Endpoint 6 transmit length register | 0x0000 |
| R16_UEP6_CTRL_H | 0x4002346A | Endpoint 6 control register | 0x0000 |
| R32_UEP7_CTRL | 0x4002346C | Endpoint 7 transmit length and control registers | 0x00000000 |
| R16_UEP7_TX_LEN | 0x4002346C | Endpoint 7 transmit length register | 0x0000 |
| R16_UEP7_CTRL_H | 0x4002346E | Endpoint 7 control register | 0x0000 |
| R32_UEPX_MOD | 0x40023470 | Endpoint X control register | 0x00000000 |

#### 18.2.2.1 USB Device Physical Port Control Register (R8_UDEV_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UD_PD_DIS | RW | USB device port UDP/UDM pin internal pull-down resistor control bits:<br>1: Disable the internal pull-down;<br>0: Enable internal pull-down, also used in GPIO mode to provide pull-down resistor.<br>*Note: MODE and CNF of GPIOC_CFGXR and GPIOC_OUTDR have been changed to set the pull-down, this bit is reserved.* | 1 |
| 6 | Reserved | RO | Reserved. | 0 |
| 5 | RB_UD_DP_PIN | RO | Current UDP pin status:<br>1: High;<br>0: low. | X |
| 4 | RB_UD_DM_PIN | RO | Current UDM pin state:<br>1: High;<br>0: low. | X |
| 3 | Reserved | RO | Reserved. | 0 |
| 2 | RB_UD_LOW_SPEED | RW | USB device physical port low speed mode enable bit:<br>1: Select 1.5Mbps low speed mode;<br>0: Select 12Mbps full speed mode. | 0 |
| 1 | RB_UD_GP_BIT | RW | USB device mode general flag bit, user-defined. | 0 |
| 0 | RB_UD_PORT_EN | RW | USB device physical port enable bit:<br>1: Enable the physical port;<br>0: Disable the physical port. | 0 |

#### 18.2.2.2 Start Address of Endpoint 0 Buffer (R32_UEP0_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| [14:0] | EP0_BUF_ADDR | RW | Start address of the endpoint 0 buffer.<br>The lower 15 bits are valid and the address must be 4-byte aligned. | 0 |

*Note: The transceiver enable signal for endpoint 0 is always active.*

#### 18.2.2.3 Start Address of Endpoint 1 Buffer (R32_UEP1_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| [14:0] | EP0_BUF_ADDR | RW | Start address of the endpoint 1 buffer.<br>The lower 15 bits are valid and the address must be 4-byte aligned. | 0 |

*Note: The transceiver enable signal for endpoint 1 is always active.*

#### 18.2.2.4 Start Address of Endpoint 2 Buffer (R32_UEP2_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| [14:0] | EP0_BUF_ADDR | RW | Start address of the endpoint 2 buffer.<br>The lower 15 bits are valid and the address must be 4-byte aligned. | 0 |

*Note: The transceiver enable signal for endpoint 2 is always active.*

#### 18.2.2.5 Start Address of Endpoint 3 Buffer (R32_UEP3_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| [14:0] | EP0_BUF_ADDR | RW | Start address of the endpoint 3 buffer.<br>The lower 15 bits are valid and the address must be 4-byte aligned. | 0 |

*Note: The transceiver enable signal for endpoint 3 is always active.*

#### 18.2.2.6 Endpoint 0 Transmit Length and Control Registers (R32_UEP0_CTRL)

| Bit | Name |
|---|---|
| [31:16] | R16_UEP0_CTRL_H |
| [15:0] | R16_UEP0_TX_LEN |

#### 18.2.2.7 Endpoint 0 transmit length register (R16_UEP0_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:0] | EP0_T_LEN | RW | The number of bytes transmitted by endpoint 0. | 0 |

#### 18.2.2.8 Endpoint 0 Control Register (R16_UEP0_CTRL_H)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | EP0_R_TOG | RW | Sync trigger bits prepared by the receiver of USB endpoint 0 (Processing OUT | 0 |

| | | | | |
|---|---|---|---|---|
| | | | transactions):<br>1: Expects DATA1;<br>0: Expects DATA0.<br>Not valid for real-time/synchronous transfers. | |
| 6 | EP0_T_TOG | RW | Sync trigger bits prepared by the sender of USB endpoint 0 (Handling IN transactions):<br>1: Transmit DATA1;<br>0: Transmit DATA0. | 0 |
| [5:4] | Reserved | RO | Reserved. | 0 |
| [3:2] | EP0_R_RES[1:0] | RW | The receiver at endpoint 0 controls the response to an OUT transaction:<br>00: Data ready and expects ACK;<br>01: Answer NYET;<br>10: Response NAK or busy;<br>11: Answer STALL or error.<br>Not valid for real-time/synchronous transfers. | 0 |
| [1:0] | EP0_T_RES[1:0] | RW | Endpoint 0 transmitter response control bits for IN transactions<br>00: Expect an answer ACK;<br>01: Expect an answer NYET;<br>10: Expect an answer NAK or busy;<br>11: Expect an answer STALL or error.<br>Not valid for real-time/synchronous transmission. | 0 |

### 18.2.2.9 Endpoint 1 Transmit Length and Control Registers (R32_UEP1_CTRL)

| Bit | Name |
|---|---|
| [31:16] | R16_UEP1_CTRL_H |
| [15:0] | R16_UEP1_TX_LEN |

### 18.2.2.10 Endpoint 1 Transmit Length Register (R16_UEP1_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:0] | EP0_T_LEN | RW | The number of bytes transmitted by endpoint 1. | 0 |

### 18.2.2.11 Endpoint 1 Control Register (R16_UEP1_CTRL_H)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | EP1_R_TOG | RW | Sync trigger bits prepared by the receiver of USB endpoint 1 (Processing OUT transactions):<br>1: Expects DATA1;<br>0: Expects DATA0.<br>Not valid for real-time/synchronous transfers. | 0 |
| 6 | EP1_T_TOG | RW | Sync trigger bits prepared by the sender of USB endpoint 1 (Handling IN transactions):<br>1: Transmit DATA1;<br>0: Transmit DATA0. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | EP1_TOG_AUTO | RW | Synchronization Trigger Bit Auto Flip Enable Control Bit:<br>1: Auto flip after successful data sending and receiving; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: Not auto-flip, can be switched manually.<br>*Note: Endpoint 0 This bit is reserved.* | |
| [3:2] | EP1_R_RES[1:0] | RW | The receiver at endpoint 1 controls the response to an OUT transaction:<br>00: Data ready and expects ACK;<br>10: Response NAK or busy;<br>11: Answer STALL or error;<br>01: Answer NYET.<br>Not valid for real-time/synchronous transfers. | 0 |
| [1:0] | EP1_T_RES[1:0] | RW | Endpoint 1 transmitter response control bits for IN transactions<br>00: Expect an answer ACK;<br>10: Expect an answer NAK or busy;<br>11: Expect an answer STALL or error;<br>01: Expect an answer NYET.<br>Not valid for real-time/synchronous transmission. | 0 |

### 18.2.2.12 Endpoint 2 Transmit Length and Control Registers (R32_UEP2_CTRL)

| Bit | Name |
|---|---|
| [31:16] | R16_UEP2_CTRL_H |
| [15:0] | R16_UEP2_TX_LEN |

### 18.2.2.13 Endpoint 2 Transmit Length Register (R16_UEP2_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:0] | EP0_T_LEN | RW | The number of bytes transmitted by endpoint 2. | 0 |

### 18.2.2.14 Endpoint 2 Control Register (R16_UEP2_CTRL_H)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | EP2_R_TOG | RW | Sync trigger bits prepared by the receiver of USB endpoint 2 (Processing OUT transactions):<br>1: Expects DATA1;<br>0: Expects DATA0;<br>Not valid for real-time/synchronous transfers. | 0 |
| 6 | EP2_T_TOG | RW | Sync trigger bits prepared by the sender of USB endpoint 2 (Handling IN transactions):<br>1: Transmit DATA1;<br>0: Transmit DATA0. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | EP2_TOG_AUTO | RW | Synchronization Trigger Bit Auto Flip Enable Control Bit:<br>1: Auto flip after successful data sending and receiving;<br>0: Not auto-flip, can be switched manually.<br>*Note: Endpoint 0 This bit is reserved.* | 0 |
| [3:2] | EP2_R_RES[1:0] | RW | The receiver at endpoint 2 controls the response to an OUT transaction:<br>00: Data ready and expects ACK;<br>10: Response NAK or busy;<br>11: Answer STALL or error; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 01: Answer NYET.<br>Not valid for real-time/synchronous transfers. | |
| [1:0] | EP2_T_RES[1:0] | RW | Endpoint 2 transmitter response control bits for IN transactions<br>00: Expect an answer ACK;<br>10: Expect an answer NAK or busy;<br>11: Expect an answer STALL or error;<br>01: Expect an answer NYET.<br>Not valid for real-time/synchronous transmission. | 0 |

### 18.2.2.15 Endpoint 3 Transmit Length and Control Registers (R32_UEP3_CTRL)

| Bit | Name |
|---|---|
| [31:16] | R16_UEP3_CTRL_H |
| [15:0] | R16_UEP3_TX_LEN |

### 18.2.2.16 Endpoint 3 Transmit Length Register (R16_UEP3_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:0] | EP0_T_LEN | RW | The number of bytes transmitted by endpoint 3. | 0 |

### 18.2.2.17 Endpoint 3 Control Register (R16_UEP3_CTRL_H)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | EP3_R_TOG | RW | Sync trigger bits prepared by the receiver of USB endpoint 3 (Processing OUT transactions):<br>1: expects DATA1;<br>0: expects DATA0.<br>Not valid for real-time/synchronous transfers. | 0 |
| 6 | EP3_T_TOG | RW | Sync trigger bits prepared by the sender of USB endpoint 3 (Handling IN transactions):<br>1: Transmit DATA1;<br>0: Transmit DATA0. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | EP3_TOG_AUTO | RW | Synchronization Trigger Bit Auto Flip Enable Control Bit:<br>1: Auto flip after successful data sending and receiving;<br>0: Not auto-flip, can be switched manually.<br>*Note: Endpoint 0 This bit is reserved.* | 0 |
| [3:2] | EP3_R_RES[1:0] | RW | The receiver at endpoint 3 controls the response to an OUT transaction:<br>00: Data ready and expects ACK;<br>01: Answer NYET;<br>10: Response NAK or busy;<br>11: Answer STALL or error.<br>Not valid for real-time/synchronous transfers. | 0 |
| [1:0] | EP3_T_RES[1:0] | RW | Endpoint 3 transmitter response control bits for IN transactions<br>00: Expect an answer ACK;<br>01: Expect an answer NYET;<br>10: Expect an answer NAK or busy; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 11: Expect an answer STALL or error.<br>Not valid for real-time/synchronous transmission. | |

### 18.2.2.18 Endpoint 4 Transmit Length and Control Registers (R32_UEP4_CTRL)

| Bit | Name |
|---|---|
| [31:16] | R16_UEP4_CTRL_H |
| [15:0] | R16_UEP4_TX_LEN |

### 18.2.2.19 Endpoint 4 Transmit Length Register (R16_UEP4_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:0] | EP4_T_LEN | RW | The number of bytes transmitted by endpoint 4. | 0 |

### 18.2.2.20 Endpoint 4 Control Register (R16_UEP4_CTRL_H)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | EP4_R_TOG | RW | Sync trigger bits prepared by the receiver of USB endpoint 4 (Processing OUT transactions):<br>1: expects DATA1;<br>0: expects DATA0.<br>Not valid for real-time/synchronous transfers. | 0 |
| 6 | EP4_T_TOG | RW | Sync trigger bits prepared by the sender of USB endpoint 4 (Handling IN transactions):<br>1: Transmit DATA1;<br>0: Transmit DATA0. | 0 |
| [5:4] | Reserved | RO | Reserved. | 0 |
| [3:2] | EP4_R_RES[1:0] | RW | The receiver at endpoint 4 controls the response to an OUT transaction:<br>00: Data ready and expects ACK;<br>01: Answer NYET;<br>10: Response NAK or busy;<br>11: Answer STALL or error.<br>Not valid for real-time/synchronous transfers. | 0 |
| [1:0] | EP4_T_RES[1:0] | RW | Endpoint 4 transmitter response control bits for IN transactions<br>00: Expect an answer ACK;<br>01: Expect an answer NYET;<br>10: Expect an answer NAK or busy;<br>11: Expect an answer STALL or error.<br>Not valid for real-time/synchronous transmission. | 0 |

### 18.2.2.21 Start Address of Endpoint 5 Buffer (R32_UEP5_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| [14:0] | EP5_BUF_ADDR | RW | Start address of the endpoint 5 buffer.<br>The lower 15 bits are valid and the address must be 4-byte aligned. | 0 |

*Note: The transceiver enable signal for endpoint 5 is always active.*

#### 18.2.2.22 Start Address of Endpoint 6 Buffer (R32_UEP6_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| [14:0] | EP6_BUF_ADDR | RW | Start address of the endpoint 6 buffer.<br>The lower 15 bits are valid and the address must be 4-byte aligned. | 0 |

*Note: The transceiver enable signal for endpoint 6 is always active.*

#### 18.2.2.23 Start Address of Endpoint 7 Buffer (R32_UEP7_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RO | Reserved. | 0 |
| [14:0] | EP7_BUF_ADDR | RW | Start address of the endpoint 7 buffer.<br>The lower 15 bits are valid and the address must be 4-byte aligned. | 0 |

*Note: The transceiver enable signal for endpoint 7 is always active.*

#### 18.2.2.24 Endpoint 5 Transmit Length and Control Registers (R32_UEP5_CTRL)

| Bit | Name |
|---|---|
| [31:16] | R16_UEP5_CTRL_H |
| [15:0] | R16_UEP5_TX_LEN |

#### 18.2.2.25 Endpoint 5 Transmit Length Register (R16_UEP5_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:0] | EP5_T_LEN | RW | The number of bytes transmitted by endpoint 5. | 0 |

#### 18.2.2.26 Endpoint 5 Control Register (R16_UEP5_CTRL_H)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | EP5_R_TOG | RW | Sync trigger bits prepared by the receiver of USB endpoint 5 (Processing OUT transactions):<br>1: Expects DATA1;<br>0: Expects DATA0.<br>Not valid for real-time/synchronous transfers. | 0 |
| 6 | EP5_T_TOG | RW | Sync trigger bits prepared by the sender of USB endpoint 5 (Handling IN transactions):<br>1: Transmit DATA1;<br>0: Transmit DATA0. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | EP5_TOG_AUTO | RW | Synchronization Trigger Bit Auto Flip Enable Control Bit:<br>1: Auto flip after successful data sending and receiving;<br>0: Not auto-flip, can be switched manually.<br>*Note: Endpoint 0 This bit is reserved.* | 0 |
| [3:2] | EP5_R_RES[1:0] | RW | The receiver at endpoint 5 controls the response to an OUT transaction:<br>00: Data ready and expects ACK;<br>01: Answer NYET;<br>10: Response NAK or busy; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 11: Answer STALL or error.<br>Not valid for real-time/synchronous transfers. | |
| [1:0] | EP5_T_RES[1:0] | RW | Endpoint 5 transmitter response control bits for IN transactions<br>00: Expect an answer ACK;<br>01: Expect an answer NYET;<br>10: Expect an answer NAK or busy;<br>11: Expect an answer STALL or error.<br>Not valid for real-time/synchronous transmission. | 0 |

### 18.2.2.27 Endpoint 6 Transmit Length and Control Registers (R32_UEP6_CTRL)

| Bit | Name |
|---|---|
| [31:16] | R16_UEP6_CTRL_H |
| [15:0] | R16_UEP6_TX_LEN |

### 18.2.2.28 Endpoint 6 Transmit Length Register (R16_UEP6_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:0] | EP6_T_LEN | RW | The number of bytes transmitted by endpoint 6. | 0 |

### 18.2.2.29 Endpoint 6 Control Register (R16_UEP6_CTRL_H)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | EP6_R_TOG | RW | Sync trigger bits prepared by the receiver of USB endpoint 6 (Processing OUT transactions):<br>1: expects DATA1;<br>0: expects DATA0.<br>Not valid for real-time/synchronous transfers. | 0 |
| 6 | EP6_T_TOG | RW | Sync trigger bits prepared by the sender of USB endpoint 6 (Handling IN transactions):<br>1: Transmit DATA1;<br>0: Transmit DATA0. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | EP6_TOG_AUTO | RW | Synchronization Trigger Bit Auto Flip Enable Control Bit:<br>1: Auto flip after successful data sending and receiving;<br>0: not auto-flip, can be switched manually.<br>*Note: Endpoint 0 This bit is reserved.* | 0 |
| [3:2] | EP6_R_RES[1:0] | RW | The receiver at endpoint 6 controls the response to an OUT transaction:<br>00: Data ready and expects ACK;<br>01: Answer NYET;<br>10: Response NAK or busy;<br>11: Answer STALL or error.<br>Not valid for real-time/synchronous transfers. | 0 |
| [1:0] | EP6_T_RES[1:0] | RW | Endpoint 6 transmitter response control bits for IN transactions<br>00: Expect an answer ACK;<br>01: Expect an answer NYET;<br>10: Expect an answer NAK or busy; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 11: Expect an answer STALL or error.<br>Not valid for real-time/synchronous transmission. | |

### 18.2.2.30 Endpoint 7 Transmit Length and Control Registers (R32_UEP7_CTRL)

| Bit | Name |
|---|---|
| [31:16] | R16_UEP7_CTRL_H |
| [15:0] | R16_UEP7_TX_LEN |

### 18.2.2.31 Endpoint 7 Transmit Length Register (R16_UEP7_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:7] | Reserved | RO | Reserved. | 0 |
| [6:0] | EP7_T_LEN | RW | The number of bytes transmitted by endpoint 7. | 0 |

### 18.2.2.32 Endpoint 7 Control Register (R16_UEP7_CTRL_H)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| 7 | EP7_R_TOG | RW | Sync trigger bits prepared by the receiver of USB endpoint 7 (Processing OUT transactions):<br>1: Expects DATA1;<br>0: Expects DATA0.<br>Not valid for real-time/synchronous transfers. | 0 |
| 6 | EP7_T_TOG | RW | Sync trigger bits prepared by the sender of USB endpoint 7 (Handling IN transactions):<br>1: Transmit DATA1;<br>0: Transmit DATA0. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | EP7_TOG_AUTO | RW | Synchronization Trigger Bit Auto Flip Enable Control Bit:<br>1: Auto flip after successful data sending and receiving;<br>0: not auto-flip, can be switched manually.<br>*Note: Endpoint 0 This bit is reserved.* | 0 |
| [3:2] | EP7_R_RES[1:0] | RW | The receiver at endpoint 7 controls the response to an OUT transaction:<br>00: Data ready and expects ACK;<br>01: Answer NYET;<br>10: Response NAK or busy;<br>11: Answer STALL or error.<br>Not valid for real-time/synchronous transfers. | 0 |
| [1:0] | EP7_T_RES[1:0] | RW | Endpoint 7 transmitter response control bits for IN transactions<br>00: Expect an answer ACK;<br>01: Expect an answer NYET;<br>10: Expect an answer NAK or busy;<br>11: Expect an answer STALL or error.<br>Not valid for real-time/synchronous transmission. | 0 |

### 18.2.2.33 Endpoint X Control Register (R32_UEPX_MOD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved. | 0 |
| [23:17] | EP_T_AF | RW | Transmit endpoint alternate enable:<br>1: Corresponds to alternate endpoints;<br>0: No alternate of endpoints. | 0 |
| 16 | Reserved | RO | Reserved. | 0 |
| [15:8] | EP_R_EN | RW | Endpoint (8-15) receive enable | 0 |
| [7:0] | EP_T_EN | RW | Endpoint (8-15) transmit enable | 0 |

### 18.2.3 USB Host Register Description

In USB host mode, the chip provides a set of bi-directional host endpoints, including a transmit endpoint OUT and a receive endpoint IN. The maximum length of a packet is 1024 bytes (Simultaneous transfers), supporting control transfers, interrupt transfers, bulk transfers and real-time/synchronous transfers.

The RB_UIF_TRANSFER interrupt flag is always automatically set at the end of processing for each USB transaction initiated by the host endpoint. The application can query directly or in the USB interrupt service program and analyze the interrupt flag register R8_USB_INT_FG to process each interrupt flag accordingly; and, if RB_UIF_TRANSFER is valid, then it also needs to continue to analyze the USB interrupt status register R8_USB_INT_ST, based on the current USB transfer If RB_UIF_TRANSFER is valid, then the USB interrupt status register R8_USB_INT_ST should be analyzed further and processed accordingly according to the current USB transfer transaction's answer PID identification MASK_UIS_H_RES.

If the sync trigger bit (RB_UH_R_TOG) of the IN transaction of the host receive endpoint is set in advance, then it can be judged by RB_U_TOG_OK or RB_UIS_TOG_OK whether the sync trigger bit of the currently received packet matches the sync trigger bit of the host receive endpoint, if the data is synchronous, then the data is valid; if the data is not synchronous, then the data should be discarded. After each USB transmit or receive interrupt is processed, the synchronization trigger bit of the corresponding host endpoint should be correctly modified to synchronize the next transmitted packet and to check whether the next received packet is synchronized; in addition, by setting RB_UH_T_AUTO_TOG and RB_UH_R_AUTO_TOG, the corresponding synchronization trigger bit can be automatically flipped after a successful transmit or receive. synchronization trigger bits.

The USB host token setting register R8_UH_EP_PID is used to set the endpoint number of the target device being operated and the token PID packet identification for this USB transmission transaction. the data corresponding to the SETUP token and OUT token is provided by the host transmit endpoint, the data ready to be sent is in the R16_UH_TX_DMA buffer, the length of the data ready to be sent is set in the R16 The data corresponding to the IN token is returned by the target device to the host receiving endpoint, the received data is stored in the R16_UH_RX_DMA buffer, and the received data length is stored in R16_USB_RX_LEN.

Table 18-4 Host-related registers list

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_HOST_CTRL | 0x40023401 | USB host physical port control register | 0xX0 |
| R8_HOST_EP_MOD | 0x4002340D | USB host endpoint mode control register | 0x00 |
| R16_HOST_RX_DMA | 0x40023418 | USB host receive buffer start address | 0xXXXX |
| R16_HOST_TX_DMA | 0x4002341C | USB host transmit buffer start address | 0xXXXX |
| R8_HOST_SETUP | 0x40023426 | USB host auxiliary setup register | 0x00 |
| R8_HOST_EP_PID | 0x40023428 | USB host token setup register | 0x00 |
| R8_HOST_RX_CTRL | 0x4002342A | USB host receive endpoint control register | 0x00 |
| R8_HOST_TX_LEN | 0x4002342C | USB host transmit length register | 0xXX |
| R8_HOST_TX_CTRL | 0x4002342E | USB host transmit endpoint control register | 0x00 |

#### 18.2.3.1 USB Host Control Register (R8_HOST_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 7 | RB_UH_PD_DIS | RW | USB host port UDP/UDM pin internal pull-down resistor control bits:<br>1: Disable the internal pull-down;<br>0: Enable internal pull-down, also used in GPIO mode to provide pull-down resistor.<br>*Note: MODE and CNF of GPIOC_CFGXR and GPIOC_OUTDR have been changed to set the pull-down, this bit is reserved* | 1 |
| 6 | Reserved | RO | Reserved. | 0 |
| 5 | RB_UH_DP_PIN | RO | Current UDP pin status:<br>1: High level;<br>0: Low level. | X |
| 4 | RB_UH_DM_PIN | RO | Current UDM pin state:<br>1: High;<br>0: Low. | X |
| 3 | Reserved | RO | Reserved. | 0 |
| 2 | RB_UH_LOW_SPEED | RW | USB host port low speed mode enable bit:<br>1: Select 1.5Mbps low-speed mode;<br>0: Select 12Mbps full-speed mode. | 0 |
| 1 | RB_UH_BUS_RESET | RW | USB host mode bus reset control bit:<br>1: Force output USB bus reset;<br>0: End output. | 0 |
| 0 | RB_UH_PORT_EN | RW | USB host port enable bit:<br>1: Enable the host port;<br>0: Disable the host port.<br>This is automatically cleared to 0 when the USB device is disconnected. | 0 |

#### 18.2.3.2 USB Host Endpoint Mode Control Register (R8_HOST_EP_MOD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved. | 0 |
| 6 | RB_UH_EP_TX_EN | RW | Host sends endpoint transmit (SETUP/OUT) enable bits:<br>1: Enable endpoint transmission;<br>0: Disable endpoint transmission. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | RB_UH_EP_TBUF_MOD | RW | Host transmit endpoint transmit data buffer mode control bit. | 0 |
| 3 | RB_UH_EP_RX_EN | RW | Host receive endpoint receive (IN) enable bit:<br>1: Enable endpoint reception;<br>0: Disable endpoint reception. | 0 |
| [2:1] | Reserved | RO | Reserved. | 00b |
| 0 | RB_UH_EP_RBUF_MOD | RW | USB host receive endpoint receive data buffer mode control bit. | 0 |

The host transmit endpoint data buffer mode is controlled by a combination of RB_UH_EP_TX_EN and RB_UH_EP_TBUF_MOD, refer to the table below.

Table 18-5 Host transmit buffer mode

| RB_UH_EP_TX_EN | RB_UH_EP_TBUF_MOD | Description: R16_UH_TX_DMA as starting address |
|---|---|---|
| 0 | X | Endpoint is disabled and the R16_UH_TX_DMA buffer is not used. |
| 1 | 0 | Single 64-byte transmit buffer (SETUP/OUT) |

| | | |
|---|---|---|
| 1 | 1 | Double 64-byte transmit buffer, selected via RB_UH_T_TOG:<br>Selection of the first 64-byte buffer when RB_UH_T_TOG = 0;<br>Back 64-byte buffer selected when RB_UH_T_TOG=1. |

The host receive endpoint data buffer mode is controlled by the combination of RB_UH_EP_RX_EN and RB_UH_EP_RBUF_MOD, refer to the table below.

Table 18-6 Host receive buffer mode

| RB_UH_EP_RX_EN | RB_UH_EP_RBUF_MOD | Structure description: R16_UH_TX_DMA as starting address |
|---|---|---|
| 0 | X | Endpoint is disabled and the R16_UH_RX_DMA buffer is not used. |
| 1 | 0 | Single 64-byte receive buffer (IN). |
| 1 | 1 | Double 64-byte receive buffer, selected via RB_UH_R_TOG:<br>Selection of the first 64 byte buffer when RB_UH_R_TOG = 0;<br>Back 64-byte buffer selected when RB_UH_R_TOG = 1. |

#### 18.2.3.3 USB Host Receive Buffer Start Address (R16_HOST_RX_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_UH_RX_DMA | RW | Start address of the host endpoint data receive buffer.<br>The lower 15 bits are valid and the address must be 4-byte aligned. | XXXXh |

#### 18.2.3.4 USB Host Transmit Buffer Start Address (R16_HOST_TX_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_UH_TX_DMA | RW | Start address of the host endpoint data transmit buffer.<br>The lower 15 bits are valid and the address must be 4-byte aligned. | XXXXh |

#### 18.2.3.5 USB Host Auxiliary Setup Register (R8_HOST_SETUP)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UH_PRE_PID_EN | RW | Low-speed leading packet PRE PID enable bit:<br>1: Enabled for communication with low-speed USB devices via the external HUB.<br>0: Disable the low-speed preamble. | 0 |
| 6 | RB_UH_SOF_EN | RW | Automatic SOF packet generation enable bit:<br>1: The host automatically generates SOF packets;<br>0: Not automatically generated, but can be generated manually. | 0 |
| [5:0] | Reserved | RO | Reserved. | 000000b |

#### 18.2.3.6 USB Host Token Setup Register (R8_HOST_EP_PID)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:4] | MASK_UH_TOKEN | RW | Set the token PID packet identifier for this | 0000b |

| | | | | |
|---|---|---|---|---|
| | | | USB transfer transaction. | |
| [3:0] | MASK_UH_ENDP | RW | Set the endpoint number of the target device being operated on this time. | 0000b |

### 18.2.3.7 USB Host Receive Endpoint Control Register (R8_HOST_RX_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UH_R_TOG | RW | Desired synchronization trigger bits for USB host receivers (Handling IN transactions):<br>1: Expect DATA1;<br>0: Expect DATA0. | 0 |
| [6:5] | Reserved | RO | Reserved. | 00b |
| 4 | RB_UH_R_AUTO_TOG | RW | Sync trigger bit auto-flip enable control bit:<br>1: Automatically flips the corresponding desired sync trigger bit (RB_UH_R_TOG) upon successful data reception;<br>0: Not automatically flipped, can be manually toggled. | 0 |
| 3 | Reserved | RO | Reserved. | 0 |
| 2 | RB_UH_R_RES | RW | Control bits for host receiver response to IN transactions:<br>1: No response, for real-time/synchronous transmission of non-zero endpoints;<br>0: Answer to ACK. | 0 |
| [1:0] | Reserved | RO | Reserved. | 00b |

### 18.2.3.8 USB Host Transmit Length Register (R8_HOST_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_UH_TX_LEN | RW | Sets the number of bytes of data that the USB host sending endpoint is ready to transmit. | XXh |

### 18.2.3.9 USB Host Transmit Endpoint Control Register (R8_HOST_TX_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved. | 0 |
| 6 | RB_UH_T_TOG | RW | Sync trigger bits prepared by the USB host sender (which handles SETUP/OUT transactions):<br>1: Indicates that DATA1 is sent;<br>0: Indicates that DATA0 is sent. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | RB_UH_T_AUTO_TOG | RW | Synchronous trigger bit auto-flip enable control bit:<br>1: The corresponding synchronous trigger bit (RB_UH_T_TOG) is automatically flipped upon successful data transmission;<br>0: Not automatically flipped, can be switched manually. | 0 |
| [3:1] | Reserved | RO | Reserved. | 000b |
| 0 | RB_UH_T_RES | RW | USB host transmitter response control bits for SETUP/OUT transactions:<br>1: No response expected, for real- | 0 |

| | | | | |
|---|---|---|---|---|
| | | | time/synchronous transmission of non-0 endpoints;<br>0: Expect an answer ACK. | |

*Note: The duration of the reset is determined by the duration of the RB_UH_TX_BUS_RST high (At least 10ms is recommended, wait a few us after the end of the reset and you can query the speed type directly). If the host wakes up the device, the hardware automatically sends a 30ms wake-up signal (K) after bUH_TX_BUS_RESUME is pulled high. bUH_TX_BUS_RESUME needs to be cleared manually to avoid affecting the next host hang-up (bUH_TX_BUS_RESUME is maintained high for at least 50ns).*

# Chapter 19 Electronic Signature (ESIG)

The electronic signature contains chip identification information: The capacity of the flash memory area and the unique identification. It is programmed into the system storage area of the memory module by the manufacturer at the factory, and can be read by SWD (SDI) or application code.

## 19.1 Functional Description

Flash memory area capacity: Indicates the available size of the current chip user application program.
Unique ID: 96-bit binary code, unique to any microcontroller; users can only read and access but cannot modify it. This unique identification information can be used as the security password, encryption key, product serial number, etc. of the microcontroller (Product) to improve the system security mechanism or indicate identity information.
Users of the above content can conduct read access according to 8/16/32 bits.

## 19.2 Register Description

Table 19-1 ESIG registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_ESIG_FLACAP | 0x1FFFF7E0 | Flash capacity register | 0xXXXX |
| R32_ESIG_UNIID1 | 0x1FFFF7E8 | UID register 1 | 0xXXXXXXXX |
| R32_ESIG_UNIID2 | 0x1FFFF7EC | UID register 2 | 0xXXXXXXXX |
| R32_ESIG_UNIID3 | 0x1FFFF7F0 | UID register 3 | 0xXXXXXXXX |

### 19.2.1 Flash Capacity Register (ESIG_FLACAP)

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F_SIZE[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | F_SIZE | RO | Flash memory capacity in Kbyte.<br>Example: 0x0080 = 128Kbytes | X |

### 19.2.2 UID Register (ESIG_UNIID1)

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| U_ID[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| U_ID[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | U_ID[31:0] | RO | UID's bits 0-31. | X |

### 19.2.3 UID Register (ESIG_UNIID2)

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| U_ID[63:48] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| U_ID[47:32] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | U_ID[63:32] | RO | UID's bits 32-63. | X |

### 19.2.4 UID Register (ESIG_UNIID3)

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| U_ID[95:80] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| U_ID[79:64] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | U_ID[95:64] | RO | UID's bits 64-95. | X |

# Chapter 20 Flash Memory and User Option Bytes

## 20.1 Flash Memory Organization

The internal flash memory organization structure of the chip is as follows:

Table 20-1 Flash memory organization structure

| Block | Name | Address Range | Size(byte) |
|---|---|---|---|
| Main memory | Page 0 | 0x0800 0000 – 0x0800 00FF | 256 |
| | Page 1 | 0x0800 0100 – 0x0800 01FF | 256 |
| | Page 2 | 0x0800 0200 – 0x0800 02FF | 256 |
| | Page 3 | 0x0800 0300 – 0x0800 03FF | 256 |
| | … | … | … |
| | Page 247 | 0x0800 F700– 0x0800 F7FF | 256 |
| Information block | Launcher code | 0x1FFF 0000 – 0x1FFF 0CFF | 3K+256 |
| | User selection word | 0x1FFF F800 – 0x1FFF F8FF | 256 |
| | Manufacturer configuration word | 0x1FFF F700 – 0x1FFF F7FF | 256 |

*Notes:*

*1) The main memory area above is used for user's application storage and is write-protected in 2K byte (8 pages) units;*

*All areas are user-operable under certain conditions, except for the "vendor configuration word" area, which is factory locked and inaccessible to the user.*

## 20.2 Flash Memory Programming and Safety

### 20.2.1 Program/Erase Methods

- Fast programming: this method uses the page operation method (Recommended). After a specific sequence of unlocking, a single 256 byte programming and 256 byte erasure, 1K byte erasure and whole chip erasure are performed.

### 20.2.2 Security - Protection against Illegal Access (Read, write, erase)

- Page write protection
- Read protection

Under the read protection state:

1） The main memory pages 0-7 (2K bytes) are automatically write-protected and are not controlled by the FLASH_WPR register; when the read protection status is released, all main memory pages will be controlled by the FLASH_WPR register.

2） The main memory cannot be erased or programmed in the system boot code area, SWD or SDI mode, and RAM area, except for the entire chip erasure. User-selected word area can be erased or programmed. If you try to release the read protection (Program user word), the chip will automatically erase the entire user area

*Note: When programming/erasing operations of flash memory are made, the internal RC oscillator (HSI) must be switched on.*

## 20.3 Register Description

Table 20-2 FLASH registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_FLASH_ACTLR | 0x40022000 | Control register | 0x00000000 |
| R32_FLASH_KEYR | 0x40022004 | FPEC key register | 0xXXXXXXXX |
| R32_FLASH_OBKEYR | 0x40022008 | OBKEY register | 0xXXXXXXXX |
| R32_FLASH_STATR | 0x4002200C | Status register | 0x0000B000 |
| R32_FLASH_CTLR | 0x40022010 | Configuration register | 0x00008080 |
| R32_FLASH_ADDR | 0x40022014 | Address register | 0x00000000 |
| R32_FLASH_OBR | 0x4002201C | Option byte register | 0x0XXXXXXX |
| R32_FLASH_WPR | 0x40022020 | Write protection register | 0xXXXXXXXX |
| R32_FLASH_MODEKEYR | 0x40022024 | Extension key register | 0xXXXXXXXX |
| R32_BOOT_MODEKEYR | 0x40022028 | Unlock BOOT key register | 0xXXXXXXXX |

### 20.3.1 Access Control Register (FLASH_ACTLR)

Offset address: 0x00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | LATENCY [1:0] | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:2] | Reserved | RO | Reserved. | 0 |
| [1:0] | LATENCY[1:0] | RW | FLASH wait state count:<br>00: 0 wait (HCLK<=12MHz);<br>01: 1 wait (12MHz<HCLK<=24MHz);<br>10: 2 wait (24MHz<HCLK<=48MHz);<br>11: Invalid. | 0 |

### 20.3.2 FPEC Key Register (FLASH_KEYR)

Offset address: 0x04

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KEYR[31:16] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KEYR[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | KEYR[31:0] | WO | FPEC keys, unlock keys for entering FPEC include:<br>KEY1 = 0x45670123;<br>KEY2 = 0xCDEF89AB. | X |

### 20.3.3 OBKEY Register (FLASH_OBKEYR)

Offset address: 0x08

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OBKEYR[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| OBKEYR[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | OBKEYR[31:0] | WO | Option byte key for entering the select word key to unlock OBWRE.<br>KEY1 = 0x45670123;<br>KEY2 = 0xCDEF89AB.<br>*(Note: Need to unlock FLASH first)* | X |

### 20.3.4 Status Register (FLASH_STATR)

Offset address: 0x0C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| LOCK | MODE | STATUS | AVA | Reserved | | | | TURBO | FWAKE_FLAG | EOP | WRPRTERR | Reserved | | | BSY |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved. | 0 |
| 15 | BOOT_LOCK | RW | BOOT zone lockout:<br>1: Locked;<br>0: Unlocked. | 1 |
| 14 | BOOT_MODE | RW | In combination with BOOT_AVA you can control the switching between the user area and the BOOT area<br>1: Switch to BOOT zone after software reset;<br>0: Switch to the user area after a software reset. | 0 |
| 13 | BOOT_STATUS | RO | Source of the currently executing program:<br>1: Indicate the program loaded from the BOOT area | 1 |
| 12 | BOOT_AVA | RO | Initial configuration word status:<br>1: Indicate boot from the BOOT area | 1 |
| [11:8] | Reserved | RO | Reserved. | 0 |
| 7 | TURBO | RO | TURBO mode enable<br>1: Indicate in TURBO mode;<br>0: No effect. | 0 |
| 6 | FWAKE_FLAG | RW0 | FLASH wake-up flag, write 0 to clear<br>1: FALSH is woken up;<br>0: No effect. | 0 |
| 5 | EOP | RW1 | Indicate end of operation, write 1 clears.<br>Set by hardware on each successful erase or programming. | 0 |
| 4 | WRPRTERR | RW1 | Indicate a write protect error, write 1 cleared. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | Hardware is set if a write protected address is programmed. | |
| [3:1] | Reserved | RO | Reserved. | 0 |
| 0 | BSY | RO | Indicate busy status:<br>1: Indicate that a flash operation is in progress;<br>0: Operation complete. | 0 |

*Note: When performing the programming operation, you need to make sure that the STRT bit in the FLASH_CTLR register is set to 0.*

### 20.3.5 Control Register (FLASH_CTLR)

Offset address: 0x10

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | BANKERASE | Reserved | | | BUFRST | BUFLOAD | FTER | FTPG |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FLOCK | Reserved | FWAKEIE | EOPIE | Reserved | ERRIE | OBWRE | Reserved | LOCK | STRT | OBER | Reserved | | MER | PER | Reserved |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved. | 0 |
| 23 | BER32 | RW | Execute block erasure 32KB<br>*Note: Only erase 0x08000000-0x08007FFF address.* | 0 |
| [22:20] | Reserved | RO | Reserved. | 0 |
| 19 | BUFRST | RW | Buffer BUF reset operation | 0 |
| 18 | BUFLOAD | RW | Cache data into BUF | 0 |
| 17 | FTER | RW | Perform fast page (256 Byte) erase operation. | 0 |
| 16 | FTPG | RW | Perform fast page programming operation. | 0 |
| 15 | FLOCK | RW1 | Fast programming lock. Write '1' only. When this bit is '1' it indicates that fast programming/erase mode is not available. Hardware clears this bit to '0' after the correct unlock sequence has been detected.<br>Software sets to 1 and re-locks. | 1 |
| 14 | Reserved | RO | Reserved. | 0 |
| 13 | FWAKEIE | RW | Wake-up interrupt enable, a flag bit is generated when the FLASH wakes up from low power mode, if the wake-up interrupt enable is set an interrupt signal is generated, otherwise it has no effect. | 0 |
| 12 | EOPIE | RW | Operation completion interrupt control (EOP set in FLASH_STATR register)<br>1: Interrupt generation is allowed;<br>0: Interrupt generation is disabled. | 0 |
| 11 | Reserved | RO | Reserved. | 0 |
| 10 | ERRIE | RW | Error status interrupt control (PGERR/WRPRTERR set in FLASH_STATR register):<br>1: Interrupt generation is allowed;<br>0: Interrupt generation is disabled. | 0 |
| 9 | OBWRE | RW0 | User select word lock, software clear 0: | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: Indicates that the user-selected word can be programmed for operation. Needs to be set by hardware after writing the correct sequence in the FLASH_OBKEYR register.<br>0: Software clear to re-lock the user select word. | |
| 8 | Reserved | RO | Reserved. | 0 |
| 7 | LOCK | RW1 | Lock. Only a '1' can be written. When this bit is '1' it means that FPEC and FLASH_CTLR are locked and unwritable. The hardware clears this bit to '0' after a correct unlock sequence is detected.<br>After an unsuccessful unlock operation, this bit will not change again until the next system reset. | 1 |
| 6 | STRT | RW1 | Start. Set to 1 to initiate an erase action and the hardware automatically clears 0 (BSY becomes '0'). | 0 |
| 5 | OBER | RW | Perform user select word erase | 0 |
| [4:3] | Reserved | RO | Reserved. | 0 |
| 2 | MER | RW | Performs a full erase operation (Erases the entire user area). | 0 |
| 1 | PER | RW | Perform a sector erase | 0 |
| 0 | Reserved | RO | Reserved. | 0 |

### 20.3.6 Address Register (FLASH_ADDR)

Offset address: 0x14

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FAR[31:16] | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FAR[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | FAR | WO | The flash address, when programming is performed, is the programmed address, and when erasing is performed, is the start address of the erase.<br>When the BSY bit in the FLASH_STATR register is '1', this register cannot be written. | 0 |

### 20.3.7 Option Byte Register (FLASH_OBR)

Offset address: 0x1C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | DATA1 | | | | | | | | DATA0 | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DATA0 | | | | | | FIX_11 | Reserved | RST_MODE | | | STANDY_RST | STOP_RST | IWDG SW | RDPRT | OBERR |

| Bit | Name | | Access | Description | Reset value |
|---|---|---|---|---|---|
| [31:26] | Reserved | | RO | Reserved. | 0 |
| [25:18] | DATA1 | | RO | Data byte 1 | X |
| [17:10] | DATA0 | | RO | Data byte 0 | X |
| [9:8] | FIX_11 | | RO | Fixed to 11 | 11b |
| 7 | User | Reserved | RO | Reserved. | X |
| [6:5] | | RST_MODE | RO | Configuration word reset delay time | X |
| 4 | | STANDY_RST | RO | System reset control in standby mode. | X |
| 3 | | STOP_RST | RO | System reset control in stop mode. | X |
| 2 | | IWDGSW | RO | Independent Watchdog (IWDG) hardware enable bit. | 1 |
| 1 | RDPRT | | RO | Read protection status.<br>1: Indicate that the flash memory is currently read protected. | 1 |
| 0 | OBERR | | RO | Select word error.<br>1: Indicate that the selection word and its inverse code do not match. | 0 |

*Note: USER and RDPRT are loaded from the user-selected word area after system reset.*

### 20.3.8 Write Protection Register (FLASH_WPR)

Offset address: 0x20

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WPR[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| WPR[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | WPR | RO | Flash memory write-protect status.<br>1: Write protection disabled;<br>0: Write protection active.<br>Each bit represents 2K bytes (8 pages) of memory write protection status. | X |

*Note: The WPR is loaded from the user-selected word area after a system reset.*

### 20.3.9 Extension Key Register (FLASH_MODEKEYR)

Offset address: 0x24

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MODEKEYR[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MODEKEYR[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | MODEKEYR[31:0] | WO | Enter the following sequence to unlock the fast programming/erase mode:<br>KEY1 = 0x45670123;<br>KEY2 = 0xCDEF89AB.<br>*(Note: FLASH needs to be unlocked first)* | X |

### 20.3.10 BOOT Key Register (BOOT_MODEKEYP)

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MODEKEYR[31:16] | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MODEKEYR[15:0] | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | MODEKEYR[31:0] | WO | Enter the following sequence to unlock the BOOT area<br>KEY1 = 0x45670123;<br>KEY2 = 0xCDEF89AB. | X |

## 20.4 Flash Operation Procedure

### 20.4.1 Read Operation

Direct addressing is in the general address space, and the user can access the content of the flash memory module and get the corresponding data through any read operation of 8/16/32-bit data.

### 20.4.2 Unlock Flash Memory

After the system reset, the flash memory controller (FPEC) and FLASH_CTLR register will be locked and cannot be accessed. The flash memory controller module can be unlocked by writing the sequence to the FLASH_KEYR register.

Unlock sequence:

1） Write KEY1 = 0x45670123 to the FLASH_KEYR register (Must operate KEY1 first);

2） Write KEY2 = 0xCDEF89AB to the FLASH_KEYR register (Must operate KEY2 secondly).

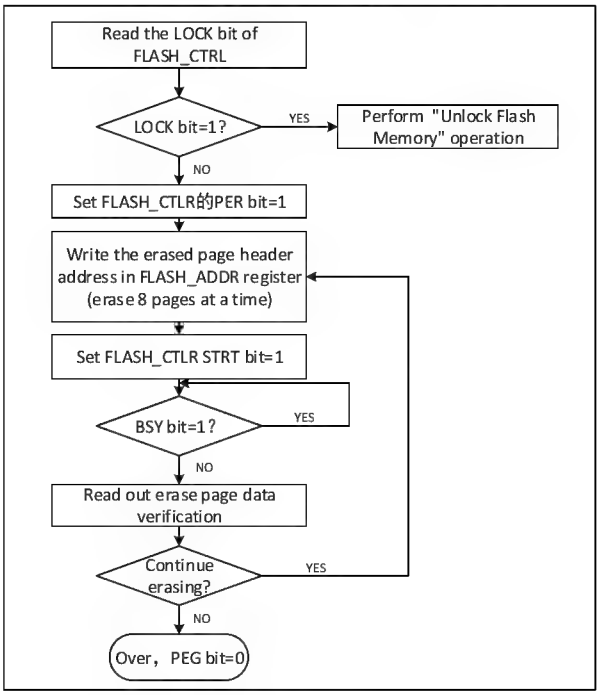
The above operations must be performed sequentially and continuously. Otherwise, it is an error operation, which will lock the FPEC module and FLASH_CTLR register and generate a bus error until the next system reset.

The flash memory controller (FPEC) and the FLASH_CTLR register can be locked again by setting the "LOCK" bit in the FLASH_CTLR register to 1.

### 20.4.3 Main Memory Standard Erase

Flash memory can be erased by standard page (1K bytes) or by whole chip.

Figure 20-1 FLASH page erase




1) Check the LOCK bit of FLASH_CTLR register, if it is 1, you need to perform the "Unlock Flash" operation.

2) Set the PER bit of FLASH_CTLR register to '1' to enable the standard page erase mode.

3) Write the page header address to the FLASH_ADDR register to select the page to be erased.

4) Set the STRT bit of FLASH_CTLR register to '1' to initiate an erase action.

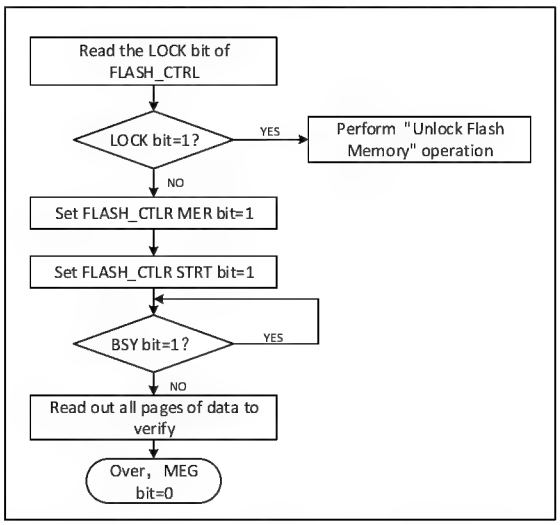
5) Wait for the BSY bit to become '0' or the EOP bit of FLASH_STATR register to become '1' to indicate the end of erasure, and clear the EOP bit to 0.

6) Read the data of the erased page for verification.

7) Continue the standard page erase can repeat steps 3-5, end the erase to clear the PEG bit to 0.

*Note: After successful erasure, the word reads - 0xFF.*

Figure 20-2 FLASH whole chip erase




1) Check the LOCK bit of FLASH_CTLR register, if it is 1, you need to perform the "unlock flash memory" operation.

2) Set the MER bit of FLASH_CTLR register to '1' to enable the whole chip erase mode.
3) Set the STRT bit of FLASH_CTLR register to '1' to start the erase operation.
4) Wait for the BSY bit to become '0' or the EOP bit of FLASH_STATR register to be '1' to indicate the end of erasure, and clear the EOP bit to 0.
5) Read the data of the erased page for verification.
6) Clear the MER bit to 0.

### 20.4.4 Fast Programming Mode Unlock

Fast programming mode operation can be unlocked by writing a sequence to the FLASH_MODEKEYR register. After unlocking, the FLOCK bit in the FLASH_CTLR register will clear to 0, indicating that fast erase and programming operations can be performed. The FLASH_CTLR register can be locked again by software setting the "FLOCK" bit to 1.
Unlock sequence:
1) Write KEY1 = 0x45670123 to the FLASH_MODEKEYR register;
2) Write KEY2 = 0xCDEF89AB to the FLASH_MODEKEYR register.
The above operations must be performed sequentially and consecutively, otherwise they will be locked as errors and cannot be unlocked until the next system reset.
*Note: The fast programming operation needs to unlock the "LOCK" and "FLOCK" layers.*

### 20.4.5 Main Memory Fast Programming

Fast programming is done on a page-by-page basis (256 bytes).
1) Check the FLASH_CTLR register LOCK bit, if it is '1', you need to perform the "Unlock Flash Memory" operation.
2) Check the FLASH_CTLR register FLOCK bit, if it is '1', you need to perform the "Fast Programming Mode Unlock" operation.
3) Check the BSY bit of the FLASH_STATR register to make sure there is no other programming operation in progress.
4) Set the FTPG bit of FLASH_CTLR register to '1' to enable fast page programming mode.
5) Set the BUFRST bit of the FLASH_CTLR register to perform a clear internal 256 byte buffer operation.
6) Wait for the BSY bit to become '0' or the EOP bit of FLASH_STATR register to become '1' to indicate the end of clearing, and clear the EOP bit to 0
7) Write data to the FLASH address using the 32-bit method, e.g.
*(uint32_t*) 0x8000000 = 0x12345678;
8) Then set the BUFLOAD bit of the FLASH_CTLR register and execute the load into the buffer.
9) Wait for BSY of FLASH_STATR register to be '0' and write the next data.
10) Repeat steps 7-9 a total of 64 times to load all 256 bytes of data into the cache (the main 64 rounds of operation addresses should be consecutive).
11) Write the first address of the fast programming page to the FLASH_ADDR register.
12) Set the STRT bit of the FLASH_CTLR register to '1' to start fast page programming.
13) Wait for the BSY bit to become '0' or the EOP bit of FLASH_STATR register to become '1' to indicate the completion of one fast page programming and clear the EOP bit to 0.
14) Query the FLASH_STATR register to see if there is an error or read the programmed address data checksum.
15) Continue fast page programming by repeating steps 5-14 and end programming by clearing the FTPG bit to 0.

### 20.4.6 Main Memory Fast Erase

Fast erase is performed by page (256 bytes).

1) Check the LOCK bit of FLASH_CTLR register, if it is 1, you need to perform the "Unlock Flash Memory" operation.

2) Check the FLASH_CTLR register FLOCK bit, if it is 1, you need to perform the "fast programming mode unlock" operation.

3) Check the BSY bit of FLASH_STATR register to make sure there is no other programming operation in progress.

4) Set the FTER bit of FLASH_CTLR register to '1' to enable the fast page erase (256 bytes) mode function.

5) Write the first address of the fast erase page to the FLASH_ADDR register.

6) Set the STRT bit of FLASH_CTLR register to '1' to start a fast page erase (256 bytes) action.

7) Wait for the BSY bit to become '0' or the EOP bit of FLASH_STATR register to become '1' to indicate the end of erase, and clear the EOP bit to 0.

8) Query the FLASH_STATR register to see if there is an error or read the erase page address data checksum.

9) Continue fast page erase can repeat steps 5-8, end erase to clear the FTER bit to 0.

*Note: After a successful erase, the word reads - 0xFF.*

Fast erase is performed by block (32K bytes).

1) Check the FLASH_CTLR register LOCK bit, if it is 1, you need to perform the "Unlock Flash" operation.

2) Check the FLASH_CTLR register FLOCK bit, if it is 1, you need to perform the "fast programming mode unlock" operation.

3) Check the BSY bit of FLASH_STATR register to make sure there is no other programming operation in progress.

4) Set the BER32 bit of FLASH_CTLR register to '1' to enable the fast block erase (32K bytes) mode function.

5) Write the first address of the fast erase block to the FLASH_ADDR register.

6) Set the STRT bit of FLASH_CTLR register to '1' to initiate a fast block erase (32K bytes) action.

7) Wait for the BSY bit to become '0' or the EOP bit of FLASH_STATR register to become '1' to indicate the end of erase, and clear the EOP bit to 0.

8) Query the FLASH_STATR register to see if there is an error or read the erase page address data checksum.

9) Continue fast page erase can repeat steps 5-8, end erase to clear the BER32 bit to 0.

*Note: After a successful erase, the word read - 0xFF.*

## 20.5 User Option Bytes

The user option byte is solidified in FLASH and is reloaded into the corresponding register after a system reset, and can be erased and programmed by the user at will. The user select word information block has a total of 8 bytes (4 bytes for write protection, 1 byte for read protection, 1 byte for configuration options and 2 bytes for user data) and each bit has its own inverse code bit for verification during loading. The structure and meaning of the select word information is described below.

Table 20-3 32-bit option byte format division

| [31:24] | [23:16] | [15:8] | [7:0] |
|---|---|---|---|
| Inverse code of option bytes 1 | Option bytes 1 | Inverse code of option bytes 0 | Option bytes 0 |

Table 20-4 User option bytes information structure

| Address / Bit | [31:24] | [23:16] | [15:8] | [7:0] |
|---|---|---|---|---|
| 0x1FFFF800 | nUSER | USER | nRDPR | RDPR |
| 0x1FFFF804 | nData1 | Data1 | nData0 | Data0 |
| 0x1FFFF808 | nWRPR1 | WRPR1 | nWRPR0 | WRPR0 |
| 0x1FFFF80C | nWRPR3 | WRPR3 | nWRPR2 | WRPR2 |

| Name/Byte | | | Description | Reset value |
|---|---|---|---|---|
| RDPR | | | Read protection control. It configures whether the code in the flash memory can be read.<br>0xA5: If this byte is 0xA5 (nRDP must be 0x5A), it means that the current code is in a non-read protected state and can be read;<br>Other values: Code read protection status, unreadable; pages 0 to 7 pages (2K) will be automatically write-protected and not controlled by WRPR0. | 0xA5 |
| USER | [7:5] | Reserved | Reserved. | xxb |
| | [4:3] | RST_MODE | External reset pin RST enable:<br>00: RST multiplexing enabled;<br>11: Multiplexing function off, PA21/PC3/PB7 for GPIO function.<br>*Note: For reset pins in different packages, refer to the datasheet for pin descriptions. For example, PA21 can be multiplexed as an external reset pin in 64-pin R8T6 and 48-pin C8T6 packages.* | 11b |
| | 2 | STANDYRST | System reset control in Standby mode:<br>1: Disable; system is not reset when entering Standby mode;<br>0: Enable; system is reset when entering Standby mode. | 1 |
| | 1 | STOPRST | System reset control in Stop mode:<br>1: Disable; system will not be reset when entering Stop mode;<br>0: Enable; system will be reset when entering Stop mode. | 1 |
| | 0 | IWDGSW | Independent watchdog (IWDG) hardware enable:<br>1: The IWDG function is enabled by software, and disabled by hardware;<br>0: The IWDG function is enabled by software (Depends on the LSI clock). | 1 |
| Data0–Data1 | | | Saving the user's data 2 bytes. | 0xFFFF |
| WRPR0 - WRPR3 | | | Write protection control. Each bit is used to control the write protection status of 2 sector (1Kbytes/sector) in main memory:<br>1: Disable write protection;<br>0: Enable write protection.<br>4 bytes are used to protect a total of 65K bytes of main memory.<br>WRP0: Sectors 0-15 memory write protection control;<br>WRP1: Sectors 16-31 memory write protection control;<br>WRP2: Sectors 32-47 memory write protection control;<br>WRP3: Bits 0-6 provide write protection for sectors 48-61; bit 7 provides write protection for sector 62 (3328 bytes of system memory). | 0xFFFFFFFF |

### 20.5.1 User Option Bytes Unlock

The user option bytes operation can be unlocked by writing the sequence to the FLASH_OBKEYR register. After unlocking, the OBWRE bit in the FLASH_CTLR register will be set to 1, indicating that user option bytes can be erased and programmed. By setting the OBWRE bit in the FLASH_CTLR register, it will be cleared to 0 by software to lock again.

Unlocking sequence

1) Write KEY1 = 0x45670123 to the FLASH_OBKEYR register;

2) Write KEY2 = 0xCDEF89AB to the FLASH_OBKEYR register.

*Note: The user needs to unlock the 2 layers: "LOCK" and "OBWRE" for word selection.*

### 20.5.2 User Option Bytes Programming

1) Check the LOCK bit of FLASH_CTLR register, if it is '1', you need to perform the operation of "Unlock Flash Memory".

2) Check the FLASH_CTLR register FLOCK bit, if it is '1', you need to perform the "Fast Programming Mode Unlock" operation.

3) Check the BSY bit of the FLASH_STATR register to make sure there is no other programming operation in progress.

4) Set the FTPG bit of FLASH_CTLR register to '1' to enable fast page programming mode.

5) Set the BUFRST bit of the FLASH_CTLR register to perform a clear internal 256 byte buffer operation.

6) Wait for the BSY bit to become '0' or the EOP bit of FLASH_STATR register to become '1' to indicate the end of clearing, and clear the EOP bit to 0

7) Write data to the FLASH address using the 32-bit method, e.g.

```
*(uint32_t*) 0x1FFFF804= 0x5AA55AA5;
```

8) Then set the BUFLOAD bit of the FLASH_CTLR register to perform a load into the buffer.

9) Wait for BSY of FLASH_STATR register to be '0' and write the next data.

10) Repeat steps 7-9 a total of 4 times to load all 16 bytes of data into the cache (The main 4 rounds of operation addresses should be consecutive). The remaining bytes of the current page use the buffer to reset the default value.

11) Write the first address of the fast programming page to the FLASH_ADDR register.

12) Set the STRT bit of the FLASH_CTLR register to '1' to start fast page programming.

13) Wait for the BSY bit to become '0' or the EOP bit of FLASH_STATR register to become '1' to indicate the completion of one fast page programming and clear the EOP bit to 0.

14) Query the FLASH_STATR register to see if there is an error or read the programmed address data checksum.

15) Continue fast page programming you can repeat steps 5-14 and end programming to clear the FTPG bit to 0.

*Note: When the "Read Protected" in the Modify Selection Word becomes "Unprotected", a whole chip erase of the main memory will be performed automatically. If a selection other than "Read Protected" is modified, a full erase operation will not occur.*

### 20.5.3 User Option Bytes Erase

Directly erase the entire 256-byte user-selected word area.

1) Check the LOCK bit in the FLASH_CTLR register, if it is 1, the "Unlock Flash" operation needs to be performed.

2) Check the BSY bit of the FLASH_STATR register to confirm that there is no programming operation in progress.

3) Check the OBWRE bit of FLASH_CTLR register, if it is 0, you need to perform the "User Option Byte

Unlock" operation.

4) Set the OBER bit of FLASH_CTLR register to '1', then set the STRT bit of FLASH_CTLR register to '1' to enable user select word erase.

5) Wait for the BSY bit to become '0' or the EOP bit of FLASH_STATR register to be '1' to indicate the end of erasure, and clear the EOP bit to 0

6) Read the erase address data checksum.

7) End to clear the OBER bit to 0.

*Note: After successful erasure, word read - 0xFF.*

### 20.5.4 Release Read Protection

Whether the flash is read protected or not is determined by the user select word. Reading the FLASH_OBR register, when the RDPRT bit is '1' indicates that the flash is currently in a read protected state and the flash is operationally protected by a series of security guards in the read protected state. The process of unprotecting a read is as follows:

1) Erase the entire user select word area, at which point the read protection field RDPR, at which point the read protection is still in effect.

2) Program the user select word to write the correct RDPR code 0xA5 to unprotect the flash memory from read protection. (This step will first cause the system to automatically perform an entire erase operation on the flash memory)

3) Perform a power-on reset to reload the selection byte (Including the new RDPR code), at which point the read protection is removed.

### 20.5.5 Release Write Protection

Whether the flash memory is write-protected or not is determined by the user select word. Read the FLASH_WPR register, each bit represents 2K bytes of flash space, when the bit is '1' it means non-write protected status, when it is '0' it means write protected. The process of unwrite-protecting is as follows:

1) Erase the entire user select word area.

2) Write the correct RDPR code 0xA5 to allow read access;

3) Perform a system reset, reload the selection byte (Including the new WRPR[3:0] byte) and the write protection is removed.

# Chapter 21 USB PD Controller (USBPD)

## 21.1 Introduction to USB PD Controller

The chip has built-in USB Power Delivery controller and PD transceiver PHY, supports USB type-C master-slave detection, automatic BMC codec and CRC, hardware edge control, supports USB PD2.0 and PD3.0 power transmission control, supports fast charging, supports UFP/DFP and PDUSB, and supports PD power receiving and PD power supply applications.

- Built-in USB type-C interface, master/slave detection support, DRP, Sink/Consumer and Source/Provider support;
- Built-in USB PD transceiver PHY with integrated hardware edge-slope control;
- Built-in USB Power Delivery controller, automatic BMC codec, 4b5b codec and CRC;
- Support SOP, SOP', SOP" PD packets and USB PD reset signal frame hardware reset;
- Support maximum packet length of 510 bytes, DMA;
- Support USB PD 2.0 and 3.0 power transfer protocols, USB port supports BC and other charging protocols.

## 21.2 Register Description

Table 21-1 USB PD-related registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R32_USBPD_CONFIG | 0x40027000 | PD configuration register | 0x00000002 |
| R16_CONFIG | 0x40027000 | PD interrupt enable register | 0x0002 |
| R16_BMC_CLK_CNT | 0x40027002 | BMC sample clock counter | 0x0000 |
| R32_USBPD_CONTROL | 0x40027004 | PD control register | 0x00000000 |
| R16_CONTROL | 0x40027004 | PD transceiver control register | 0x0000 |
| R8_CONTROL | 0x40027004 | PD transceiver enable register | 0x00 |
| R8_TX_SEL | 0x40027005 | PD transmit SOP selection register | 0x00 |
| R16_BMC_TX_SZ | 0x40027006 | PD transmit length register | 0x0000 |
| R32_USBPD_STATUS | 0x40027008 | PD status register | 0x000000XX |
| R16_STATUS | 0x40027008 | PD interrupt and data register | 0x00XX |
| R8_DATA_BUF | 0x40027008 | DMA cache data register | 0xXX |
| R8_STATUS | 0x40027009 | PD interrupt flag register | 0x00 |
| R16_BMC_BYTE_CNT | 0x4002700A | Byte counter | 0x0000 |
| R32_USBPD_PORT | 0x4002700C | Port control register | 0x00030003 |
| R16_PORT_CC1 | 0x4002700C | CC1 port control register | 0x0003 |
| R16_PORT_CC2 | 0x4002700E | CC2 port control register | 0x0003 |
| R32_USBPD_DMA | 0x40027010 | DMA cache address register | 0x0000XXXX |
| R16_DMA | 0x40027010 | PD buffer start address register | 0xXXXX |

### 21.2.1 Configuration Register (R32_USBPD_CONFIG)

Offset address: 0x00

| Bit | Name |
|---|---|
| [31:16] | R16_BMC_CLK_CNT |
| [15:0] | R16_CONFIG |

### 21.2.2 PD Interrupt Enable Register (R16_CONFIG)

Offset address: 0x00

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IE_TX_END | IE_RX_RESET | IE_RX_ACT | IE_RX_BYTE | IE_RX_BIT | IE_PD_IO | Reserved | | | | WAKE_POLAR | PD_RST_EN | PD_DMA_EN | CC_SEL | PD_ALL_CLR | PD_FILT_EN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | IE_TX_END | RW | Transmit end interrupt enable. | 0 |
| 14 | IE_RX_RESET | RW | Receive reset interrupt enable. | 0 |
| 13 | IE_RX_ACT | RW | Receive complete interrupt enable. | 0 |
| 12 | IE_RX_BYTE | RW | Receive byte interrupt enable. | 0 |
| 11 | IE_RX_BIT | RW | Receive bit interrupt enable. | 0 |
| 10 | IE_PD_IO | RW | PD IO interrupt enable. | 0 |
| [9:6] | Reserved | RO | Reserved. | 0 |
| 5 | WAKE_POLAR | RW | PD port wake-up level:<br>1: Active high;<br>0: Active low. | 0 |
| 4 | PD_RST_EN | RW | PD mode reset command enable:<br>1: Reset;<br>0: Invalid. | 0 |
| 3 | PD_DMA_EN | RW | Enable DMA for USBPD, this bit must be set to 1 in normal transmission mode:<br>1: Enable DMA function and DMA interrupt;<br>0: Disable DMA. | 0 |
| 2 | CC_SEL | RW | Select the current PD communication port:<br>1: Use the CC2 port for communication;<br>0: Use the CC1 port for communication. | 0 |
| 1 | PD_ALL_CLR | RW | PD mode clears all interrupt flag bits: 0: invalid; 1: use CC2 port<br>1: Clear the interrupt flag bits;<br>0: Invalid. | 1 |
| 0 | PD_FILT_EN | RW | Control the input filter enable on the PD pin:<br>1: Turn on filtering;<br>0: Turn off filtering. | 0 |

### 21.2.3 BMC Sample Clock Counter (R16_BMC_CLK_CNT)

Offset address: 0x02

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | BMC_CLK_CNT | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:9] | Reserved | RO | Reserved. | 0 |
| [8:0] | BMC_CLK_CNT | RW | The BMC transmits or receives a sample clock counter. | 0 |

### 21.2.4 Control Register (R32_USBPD_CONTROL)

Offset address: 0x04

| Bit | Name |
|---|---|
| [31:16] | R16_BMC_TX_SZ |
| [15:0] | R16_CONTROL |

### 21.2.5 PD Transceiver Control Register (R16_CONTROL)

Offset address: 0x04

| Bit | Name |
|---|---|
| [15:8] | R8_TX_SEL |
| [7:0] | R8_CONTROL |

### 21.2.6 PD Transceiver Enable Register (R8_CONTROL)

Offset address: 0x04

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BMC_BYTE_HI | TX_BIT_BACK | DATA_FLAG | RX_STATE | | | BMC_START | PD_TX_EN |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | BMC_BYTE_HI | RO | Indicates the current half-byte status of the PD data being sent and received:<br>1: Indicates that the high 4 bits are being;<br>0: The lower 4 bits are being processed. | 0 |
| 6 | TX_BIT_BACK | RO | Indicates the current bit status of the BMC when sending the code:<br>1: Indicate that a BMC byte is being transmitted;<br>0: Idle. | 0 |
| 5 | DATA_FLAG | RO | Cache data valid flag bit. | 0 |
| [4:2] | RX_STATE | RO | PD receive status identification<br>000: Receive initial status 001: Start receiving SOP<br>010: Receive reset 011: Receive SOP<br>100: Receive end 101: Receive unused<br>110: Receive EOP 111: Receive byte. | 0 |
| 1 | BMC_START | RW | BMC transmit start signal. | 0 |
| 0 | PD_TX_EN | RW | USBPD receive and transmit mode and transmit enable:<br>1: PD transmit enable;<br>0: PD receive enable. | 0 |

### 21.2.7 SOP Port Selection Register (R8_TX_SEL)

Offset address: 0x05

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TX_SEL4 | | TX_ SEL3 | | TX_ SEL2 | | Reserved | TX_ SEL1 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:6] | TX_SEL4 | RW | K-CODE4 type selection in PD transmit mode:<br>00: SYNC2;<br>01: SYNC3;<br>1x: RST2. | 0 |
| [5:4] | TX_SEL3 | RW | K-CODE3 type selection in PD transmit mode:<br>00: SYNC1;<br>01: SYNC3;<br>1x: RST1. | 0 |
| [3:2] | TX_SEL2 | RW | K-CODE2 type selection in PD transmit mode:<br>00: SYNC1;<br>01: SYNC3;<br>1x: RST1. | 0 |
| 1 | Reserved | RO | Reserved | 0 |

| | | | | |
|---|---|---|---|---|
| 0 | TX_SEL1 | RW | K-CODE1 type selection in PD transmit mode:<br>1: RST1;<br>0: SYNC1. | 0 |

### 21.2.8 PD Transmit Length Register (R16_BMC_TX_SZ)

Offset address: 0x06

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | BMC_TX_SZ | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:9] | Reserved | RO | Reserved. | 0 |
| [8:0] | BMC_TX_SZ | RW | The total length sent in PD mode. | 0 |

### 21.2.9 Status Register (R32_USBPD_STATUS)

Offset address: 0x08

| Bit | Name |
|---|---|
| [31:16] | R16_BMC_BYTE_CNT |
| [15:0] | R16_STATUS |

### 21.2.10 PD Interrupt and Data Register (R16_STATUS)

Offset address: 0x08

| Bit | Name |
|---|---|
| [15:8] | R8_STATUS |
| [7:0] | R8_DATA_BUF |

### 21.2.11 DMA Buffer Data Register (R8_DATA_BUF)

Offset address: 0x08

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DATA_BUF | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | DATA_BUF | RO | DMA buffer data. | X |

### 21.2.12 PD Interrupt Flag Register (R8_STATUS)

Offset address: 0x09

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IF_TX_END | IF_RX_RESET | IF_RX_ACT | IF_RX_BYTE | IF_RX_BIT | BUF_ERR | BMC_AUX | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | IF_TX_END | RW1 | Transmit complete interrupt flag, write 1 to clear 0, write 0 is invalid. | 0 |
| 6 | IF_RX_RESET | RW1 | Receive reset interrupt flag, write 1 to clear 0, write 0 is invalid. | 0 |
| 5 | IF_RX_ACT | RW1 | Receive complete interrupt flag, write 1 to clear 0, write 0 is invalid. | 0 |
| 4 | IF_RX_BYTE | RW1 | Receive byte or SOP interrupt flag, write 1 to | 0 |

| | | | | |
|---|---|---|---|---|
| | | | clear 0, write 0 is invalid. | |
| 3 | IF_RX_BIT | RW1 | Receive bit or 5bit interrupt flag, write 1 to clear 0, write 0 is invalid. | 0 |
| 2 | BUF_ERR | RW1 | BUFFER or DMA error interrupt flag, write 1 to clear 0, write 0 is invalid. | 0 |
| [1:0] | BMC_AUX | RO | Indicates the current PD status:<br>At the time of PD reception or after completion of reception, the state is as follows:<br>00: reception idle or no valid packet received;<br>01: SOP received i.e. SOP0;<br>10: SOP' received i.e. SOP1 or Hard Reset;<br>11: SOP' received i.e. SOP2 or Cable Reset.<br>When PD is sending, the status is as follows:<br>00: CRC32[7:0] is being transmitted;<br>01: CRC32[15:8] is being transmitted;<br>10: CRC32[23:16] is being transmitted;<br>11: CRC32[31:24] is being transmitted. | 00 |

### 21.2.13 Byte Counter (R16_BMC_BYTE_CNT)

Offset address: 0x0A

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | BMC_BYTE_CNT | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:9] | Reserved | RO | Reserved. | 0 |
| [8:0] | BMC_BYTE_CNT | RO | Byte counter. | 0 |

### 21.2.14 Port Control Register (R32_USBPD_PORT)

Offset address: 0x0C

| Bit | Name |
|---|---|
| [31:16] | R16_PORT_CC2 |
| [15:0] | R16_PORT_CC1 |

### 21.2.15 CC1 Port Control Register (R16_PORT_CC1)

Offset address: 0x0C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | CC1_CE | | | CC1_LV0 | CC1_PU | | Reserved | PA_CC1_AI |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| [7:5] | CC1_CE | RW | Enable the voltage comparator on port CC1:<br>000: off;<br>001: reserved;<br>010: 0.22V;<br>011: 0.43V;<br>100: 0.55V;<br>101: 0.66V;<br>110: 0.96V;<br>111: 1.23V. | 000 |
| 4 | CC1_LVE | RW | CC1 port output low voltage enable: | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: Normal VDD voltage drive output;<br>1: Low voltage drive output. | |
| [3:2] | CC1_PU | RW | CC1 port pull-up current selection:<br>00: Pull-up current disable;<br>01: 330uA;<br>10: 180uA;<br>11: 80uA. | 00 |
| 1 | Reserved | RO | Reserved. | 1 |
| 0 | PA_CC1_AI | RO | CC1 port comparator analog input. | 1 |

### 21.2.16 CC2 Port Control Register (R16_PORT_CC2)

Offset address: 0x0E

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | CC2_CE | | | CC2_LV0 | CC2_PU | | Reserved | PA_CC2_AI |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| [7:5] | CC2_CE | RW | Enable the voltage comparator on port CC2:<br>000: off;<br>001: reserved;<br>010: 0.22V;<br>011: 0.43V;<br>100: 0.55V;<br>101: 0.66V;<br>110: 0.96V;<br>111: 1.23V. | 000 |
| 4 | CC2_LVE | RW | CC2 port output low voltage enable:<br>1: Low voltage drive output;<br>0: Normal $V_{DD}$ voltage drive output. | 0 |
| [3:2] | CC2_PU | RW | CC2 port pull-up current selection:<br>00: Pull-up current disable;<br>01: 330uA;<br>10: 180uA;<br>11: 80uA. | 00 |
| 1 | Reserved | RO | Reserved. | 1 |
| 0 | PA_CC2_AI | RO | CC2 port comparator analog input. | 1 |

### 21.2.17 DMA Cache Address Register (R32_USBPD_DMA)

Offset address: 0x10

| Bit | Name |
|---|---|
| [31:16] | Reserved |
| [15:0] | R16_DMA |

### 21.2.18 PD Buffer Start Address Register (R16_DMA)

Offset address: 0x10

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| USBPD_DMA_ADDR | | | | | | | | | | | | | | | |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | USBPD_DMA_ADDR | RW | USBPD_DMA cache address. | X |

| | | | | |
|---|---|---|---|---|
| | | | The lower 16 bits are valid and the address must be 4-byte aligned. | |

# Chapter 22 Programmable Protocol I/O Microcontroller (PIOC)

The CH32X035 chip is embedded with a programmable protocol I/O microcontroller based on a single clock cycle dedicated lean instruction set RISC core running at system mains frequency with 2K instruction program ROM and 49 SFR registers and PWM timer/counter, supporting protocol control of 2 I/O pins.

## 22.1 Main Features

- RISC core, optimized single cycle bit manipulation instruction set, fully static design.
- Multiplex 4K bytes of system SRAM as 2K capacity program ROM, supports program pause and dynamic loading.
- Provide 33 bytes of 1 register each in bidirectional and unidirectional mode, providing 6 levels of independent stacking.
- Support 2 general purpose bi-directional I/O protocol controls and input level change detection.
- Support 1- and 2-wire interfaces with multiple protocol specifications by dynamically loading different protocol programs.

## 22.2 Applications

A wide range of protocol specification target programs are available, dynamically loaded to support the corresponding 1- and 2-wire interfaces.

# Chapter 23 Debug Support

## 23.1 Main Features

This register allows the MCU to be configured in the debug state. includes

- Support counter of Independent Watchdog (IWDG)
- Support counter of Window Watchdog (WWDG)
- Support counter of Timer 1
- Support counter of Timer 2
- Support counter of Timer 3

## 23.2 Register Description

### 23.2.1 Debug MCU Configuration Register (DBGMCU_CR)

Address: 0x7C0(CSR)

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | TIM3_STOP | TIM2_STOP | TIM1_STOP | Reserved | | WWDG_STOP | IWDG_STOP | Reserved | | | | | STANDBY | STOP | SLEEP |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:15] | Reserved | RW | Reserved | 0 |
| 14 | TIM3_STOP | RW | Timer 3 debug stop bit. The counter stops when the core enters the debug state.<br>1: Timer 3's counter stops working;<br>0: Timer 3's counter is still operating normally. | 0 |
| 13 | TIM2_STOP | RW | Timer 2 debug stop bit. The counter stops when the core enters the debug state.<br>1: Timer 2's counter stops working;<br>0: Timer 2's counter is still operating normally. | 0 |
| 12 | TIM1_STOP | RW | Timer 1 debug stop bit. The counter stops when the core enters the debug state.<br>1: Timer 1's counter stops working;<br>0: Timer 1's counter is still operating normally. | 0 |
| [10:11] | Reserved | RW | Reserved | 0 |
| 9 | WWDG_STOP | RW | Window watchdog debug stop bit. The debug window watchdog stops when the core enters the debug state.<br>1: The window watchdog counter stops working;<br>0: The window watchdog counter is still operating normally. | 0 |
| 8 | IWDG_STOP | RW | Independent watchdog debug stop bit. The watchdog stops when the core enters the debug state.<br>1: The watchdog counter stops working;<br>0: The watchdog counter is still working normally. | 0 |
| [7:3] | Reserved | RW | Reserved | 0 |
| 2 | STANDBY | RW | Debug the standby mode bits.<br>1: (HCLK on) The digital circuitry section is not | 0 |

| | | | | |
|---|---|---|---|---|
| | | | powered down and the HCLK clock is clocked by the internal RL oscillator. Alternatively, the microcontroller exits STANDBY mode and reset by generating a system reset is the same;<br>0: (HCLK off) The entire digital circuitry section is powered down.<br>From a software point of view, exiting STANDBY mode is the same as a reset (Except that some status bits indicate that the microcontroller has just exited from the STANDBY state). | |
| 1 | STOP | RW | Debug stop mode bit.<br>1: (HCLK on) When in stop mode, the HCLK clock is provided by the internal RC oscillator. When exiting stop mode, the software must reconfigure the clock system to start the PLL, crystal, etc. (Same operation as when this bit is configured to 0);<br>0: (HCLK off) When in STOP mode, the clock controller disables all clocks (Including HCLK). When exiting from STOP mode, the clock is configured as it was after the reset (The microcontroller is clocked by the internal RC oscillator (HSI)). Therefore, the software must reconfigure the clock control system to start the PLL, crystal, etc. | 0 |
| 0 | SLEEP | RW | Debug sleep mode bits.<br>1: (HCLK on) In sleep mode, the HCLK clocks are all provided by the originally configured system clock;<br>0: (HCLK Off) In sleep mode, HCLK is provided by the originally configured system clock and HCLK is off. Since sleep mode does not reset the configured clock system, the software does not need to reconfigure the clock system when exiting from sleep mode. | 0 |